जावास्क्रिप्ट के प्रयोग से क्लाइंट की ओर से स्क्रिप्टिंग

10

उद्देश्य

इस अध्याय को पूरा करने के बाद छात्र यह करने में सक्षम होंगे–

- *जावास्क्रिप्ट को परिभाषित करने में,*
- *जावास्क्रिप्ट के बारे में आधारभूत जानकारी के बारे में बताने में,*
- *जावास्क्रिप्ट के कोड को एचटीएमएल प्रलेख में रखने में,*
- *चरों की तुलना करने, उनके बारे में बताने और उनका प्रयोग करने में,*
- *अभिव्यक्तियों में चरों और शब्दों का प्रयोग करने में,*
- *विभिन्न प्रकार के डाटा और मानों का ब्योरा देने में,*
- *ब्रांचिंग और लूप कथनों का प्रयोग करने में,*
- *लूप **'हेतु'** के साथ पुनरुक्ति करने में,*
- ***'जबकि'** और **'करे ... जबकि'** लूप के बीच अंतर स्पष्ट करने में,*
- *लूप को तोड़ने और जारी रखने में,*
- *कुछ वस्तु कौशल कथनों पर चर्चा करने में, और*
- *कृत्यों को परिभाषित करने और उन्हें करने पर विचार करने में।*

"विषय-वस्तु निर्माण जटिल नहीं होना चाहिए। यह विचित्र/असामान्य रहस्य जैसा नहीं होना चाहिए कि केवल विशेषज्ञ और कम्प्यूटर साइंस के अति-विद्वान गुरु ही यह कार्य कर सकें।"

ब्रेंडन ईच
जावास्क्रिप्ट निर्माता

प्रस्तावना

अध्याय 9 में हमने सीखा कि एचटीएमएल के प्रयोग द्वारा वेब पेज कैसे निर्मित किया जाता है। एचटीएमएल प्रलेखों में कई टैग ऐसे होते हैं जो ब्राउज़र को बताते हैं कि टेक्स्ट या ग्राफिक्स को कैसे डिस्प्ले (प्रदर्शित) करना है। साथ ही, हमने स्थिर वेब पेज निर्मित करना भी सीखा है जो अंतरक्रियात्मक नहीं होते। एचटीएमएल प्रलेख को अंतरक्रियात्मक और गतिशील बनाने के लिए एचटीएमएल के साथ कुछ विशेष कोड (स्क्रिप्ट) को शामिल किए जाने की आवश्यकता है। ऐसी ही एक स्क्रिप्टिंग लैंग्वेज है जावास्क्रिप्ट।

इस अध्याय में हम जावास्क्रिप्ट के बारे में आधारभूत जानेंगे और सीखेंगे कि अभिव्यक्ति (expression), प्रचालकों (operators), पॉप-अप बॉक्सेज, परिवर्तन कृत्यों (conversion functions), कंडीशनल स्टेटमेंट्स, लूपिंग स्टेटमेंट्स, वस्तु-कौशल कथनों (object manipulation statements), और जावास्क्रिप्ट कृत्यों के प्रयोग से वेब पेजों में गतिज प्रभाव कैसे लाया जा सकता है।

10.1 जावास्क्रिप्ट क्या है?

जावास्क्रिप्ट ब्रेंडन ईच द्वारा निर्मित की गई थी और यह 1995 में जब नेटस्केप 2.0 (यह वेब ब्राउज़र है) को जारी किया गया था, तभी अस्तित्व में आया था। जावास्क्रिप्ट को वेब पेजों को अधिक गतिज और अंतरक्रियात्मक बनाने के उद्देश्य से तैयार किया गया था। जावास्क्रिप्ट लोकप्रिय स्क्रिप्टिंग लैंग्वेजों में से एक है और इसकी निम्नवत् विशेषताएं हैं–

(क) इसका क्लाइंट और सर्वर अनुप्रयोगों हेतु प्रयोग किया जा सकता है।

(ख) इसे किसी प्लेटफॉर्म की आवश्यकता नहीं है अर्थात् यह किसी भी ऑपरेटिंग सिस्टम (अर्थात् लिनक्स, माइक्रोसॉफ्ट विन्डोज़, मैक ओ.एस. एक्स इत्यादि) पर चलाया जा सकता है।

(ग) जावास्क्रिप्ट कोडों को वेब ब्राउज़र में केवल तभी चलाया जा सकता है जब इन्हें एचटीएमएल प्रलेख में रखा जाए या उन्हें संदर्भित किया जाए।

(घ) यह इंटरप्रेटेड व्याख्यायित लैंग्वेज है।

(ङ) यह एक केस-सेंसिटिव लैंग्वेज है तथा इसके की-वर्ड केवल लोअरकेस में होते हैं।

10.1.1 जावा और जावास्क्रिप्ट में अंतर

कुछ लोग सोचते हैं कि जावा और जावास्क्रिप्ट समान हैं लेकिन ये दोनों बिलकुल अलग-अलग लैंग्वेज हैं। जावा सन माइक्रोसिस्टम की सामान्य प्रयोजन संबंधी वस्तु उन्मुख प्रोग्रामिंग लैंग्वेज है जबकि जावास्क्रिप्ट वस्तु-आधारित स्क्रिप्टिंग लैंग्वेज है। स्क्रिप्ट से अभिप्राय है टास्क करने के लिए लघु प्रोग्रामिंग कथन।

10.1.2 जावास्क्रिप्ट के संस्करण

जावास्क्रिप्ट और वेब ब्राउज़र के कुछ संस्करण हैं–

जावास्क्रिप्ट संस्करण	*वेब ब्राउज़र*
जावास्क्रिप्ट 1.0	नेवीगेटर 2.0, इंटरनेट एक्सप्लोरर 3.0
जावास्क्रिप्ट 1.3	नेवीगेटर 4.06-4.7x, इंटरनेट एक्सप्लोरर 4.0
जावास्क्रिप्ट 1.5	नेवीगेटर 6.0, मॉजिला, इंटरनेट एक्सप्लोरर 5.5
जावास्क्रिप्ट 1.6	मॉजिला फायरफॉक्स 1.5
जावास्क्रिप्ट 1.7	मॉजिला फायरफॉक्स 2.0
जावास्क्रिप्ट 1.8	मॉजिला फायरफॉक्स 3.0

माइक्रोसॉफ्ट ने जावास्क्रिप्ट के कई संस्करण निकाले हैं और वर्तमान में इंटरनेट एक्सप्लोरर 7.0 के साथ जावास्क्रिप्ट संस्करण 5.7 का प्रयोग किया जाता है।

10.2 क्लाइंट सर्वर मॉडल

आगे बढ़ने से पहले हमें नोड, क्लाइंट और सर्वर के बारे में जानकारी होनी चाहिए।

10.2.1 नोड

नोड नेटवर्क से जुड़ा हुआ एक कंपोनेंट अथवा टर्मिनल है। कम्प्यूटर नेटवर्क में कंपोनेंट्स जैसे– लैपटॉप्स, पीडीए, इंटरनेट समर्थित मोबाइल इत्यादि को नोड माना जा सकता है।

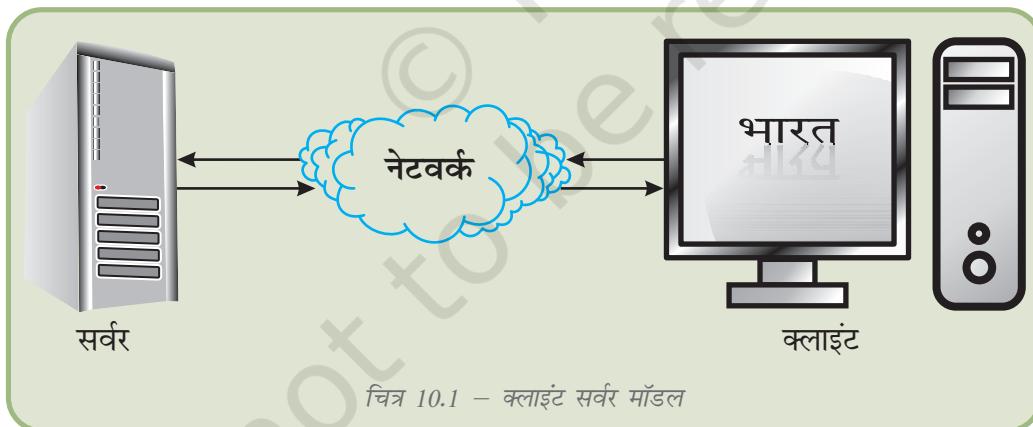
10.2.2 क्लाइंट

यह एक नोड कम्प्यूटर है जो सर्वर से संपर्क स्थापित करता है, प्रयोक्ता से डाटा एकत्रित कर इसे सर्वर को भेजता है और सर्वर से सूचना प्राप्त कर इसे प्रयोक्ता को प्रस्तुत करता है।

10.2.3 सर्वर

क्लाइंट-सर्वर मॉडल के संदर्भ में, सर्वर क्लाइंट का प्रतिपूर्ण (counter part) है। यह एक कम्प्यूटर है जो क्लाइंट से प्राप्त प्रश्नों का समाधान करता है। जो प्रोग्राम क्लाइंट के अनुरोध का उत्तर देते हैं उन्हें सर्वर अनुप्रयोग कहा जाता है। सर्वर अनुप्रयोग को चलाने के लिए तैयार किए गए कम्प्यूटर को सर्वर मशीन कहा जाता है। वेब सर्वर, डाटाबेस सर्वर और मेल सर्वर, सर्वरों के कुछ उदाहरण हैं।

क्लाइंट-सर्वर मॉडल कम्प्यूटर नेटवर्क का स्थापत्य है जिसमें नेटवर्क (चित्र 10.1) के माध्यम से क्लाइंट और सर्वर पारस्परिक क्रिया करते हैं। क्लाइंट इनपुट के रूप में प्रयोक्ता से डाटा एकत्रित करता है और सर्वर के लिए रिक्वेस्ट भेजता है। सर्वर रिक्वेस्ट (अनुरोध) पर कार्यवाही करता है और वांछित सूचना को क्लाइंट को भेजता है। रेलवे टिकट आरक्षण प्रणाली, ऑनलाइन बैंकिंग और ऑनलाइन गेमिंग क्लाइंट-सर्वर मॉडल के उदाहरण हैं।

चित्र 10.1 – क्लाइंट सर्वर मॉडल

10.2.4 क्लाइंट-साइड जावास्क्रिप्ट

क्लाइंट-साइड जावास्क्रिप्ट से आशय जावास्क्रिप्ट कोड से है जिसे क्लाइंट मशीन पर वेब ब्राउज़र द्वारा कार्यान्वित किया जाता है। जावास्क्रिप्ट कथन HTML प्रलेख के अंदर हो सकते हैं जो बिना किसी नेटवर्क कनेक्शन के माउस क्लिक, फॉर्म इनपुट और पेज नेविगेशन इत्यादि जैसे इवेंटों का उत्तर दे सकते हैं।

10.2.5 सर्वर-साइड जावास्क्रिप्ट

सर्वर-साइड जावास्क्रिप्ट को "Live Wire" भी कहा जाता है। क्लाइंट-साइड, जावास्क्रिप्ट की तरह ही सर्वर-साइड जावास्क्रिप्ट भी HTML प्रलेख के अंदर होता है। जब क्लाइंट द्वारा प्रश्न पूछा जाता है, तब वेब सर्वर इसे इंटरप्रेट करने के पश्चात् स्क्रिप्ट को कार्यान्वित करता है।

10.3 जावास्क्रिप्ट से आरंभ करना

जावास्क्रिप्ट एक पूर्ण लैंग्वेज नहीं है और इसे HTML प्रलेख के अंतर्गत रखे जाने की आवश्यकता है। अन्यथा, या जावास्क्रिप्ट कोड वाली किसी बाह्य फाइल को विनिर्दिष्ट करने के लिए हम जावास्क्रिप्ट कोड हेतु 'स्क्रिप्ट' या 'प्रोग्राम' शब्द का प्रयोग कर सकते हैं।

HTML प्रलेख में <Script> और HTML </Script> टैगों का प्रयोग स्क्रिप्ट को आरंभ करने का एक सामान्य तरीका है। हम अपने जावास्क्रिप्ट कोड को HTML प्रलेख के HEAD या BODY खंड में रख सकते हैं।

सिन्टैक्स (सामान्य फॉर्मेट) निम्न है–

<Script [Attributes = ["value"] .....]> — जावास्क्रिप्ट कोड के आरंभ होने की ओर संकेत करता है।

...JavaScript statement(s);

</SCRIPT> — जावास्क्रिप्ट कोड के समाप्त होने की ओर संकेत करता है।

निम्नलिखित सारणी में स्क्रिप्ट विशेषताएं, मान और उनका विवरण दिया गया है–

विशेषता	मान	विवरण
प्रकार	text/javascript text/ecmascript text/vbscript	स्क्रिप्ट का प्रकार
लैंग्वेज	Javascript vbscript	स्क्रिप्टिंग लैंग्वेज का नाम
Src	URL	फाइल का यूआरएल जिसमें स्क्रिप्ट है।

प्रोग्राम 10.1 – document.write ( ) के प्रयोग से पहला साधारण जावास्क्रिप्ट प्रोग्राम।

```
<HTML>
<HEAD>
<TITLE>My First JavaScript program </TITLE>
</HEAD>
<BODY>
<SCRIPT type="text/javascript">
   document.write("Welcome to JavaScript ↵
Programming!");
</SCRIPT>
</BODY>
</HTML>
```

My First JavaScript program - Mozilla Firefox
File Edit View History Bookmarks Tools Help
My First JavaScript program
Welcome to JavaScript Programming!
Done

चित्र 10.2

नोट – आप देखेंगे कि कोड एक पंक्ति में पूरा नहीं आता है। इसलिए हम यह बताने के लिए कि कोड जारी है और इसे अंतर्विष्ट करने के लिए आपको टाइप करने की आवश्यकता नहीं है, हमने ↵ का प्रयोग किया।

उपर्युक्त प्रोग्राम को चलाने के लिए, नोटपैड, वर्डपैड जैसे किसी भी टेक्स्ट एडीटर के प्रयोग से ऊपर दिए प्रोग्राम कोड को टाइप करें और "<file name>.htm" (उदाहरण के लिए abc.htm) के रूप में इसे सुरक्षित (सेव) करें। किसी भी ब्राउज़र अनुप्रयोग (अर्थात् इंटरनेट एक्सप्लोरर, मॉजिला फायरफॉक्स, ओपेरा इत्यादि) के प्रयोग द्वारा इस फाइल को खोल सकते हैं।

जावास्क्रिप्ट कोड को लिखने एवं चलाने हेतु आवश्यक उपकरण

जावास्क्रिप्ट कोड में कार्य करने हेतु निम्नलिखित उपकरणों की आवश्यकता होती है–

(क) **टेक्स्ट एडीटर**– हम कोई भी टेक्स्ट एडीटर या वर्ड प्रोसेसर का चयन कर सकते हैं (अर्थात् नोटपैड, वर्डपैड इत्यादि)।

(ख) **ब्राउज़र**– ब्राउज़र जावास्क्रिप्ट कोड की व्याख्या करता है और आउटपुट को ब्राउज़र के प्रलेख विंडो में दिखाता है।

10.3.1 जावास्क्रिप्ट कोड को रखना

जावास्क्रिप्ट कोड को रखने के दो तरीके हैं–

1. **एम्बेडेड/इनलाइन जावास्क्रिप्ट**– जावास्क्रिप्ट कोड को HTML प्रलेख के HEAD या BODY खंड में रख सकते हैं।
   (क) जब कोड को एक से अधिक बार प्रयोग करना हो तो जावास्क्रिप्ट कोड को HEAD खंड में रखने की सलाह दी जाती है।
   (ख) यदि जावास्क्रिप्ट कोड आकार में छोटा हो और एक बार प्रयोग करना हो तो इसे HTML प्रलेख के BODY खंड में रखें।
2. **बाह्य जावास्क्रिप्ट**– यदि बहु-प्रलेखों में समान जावास्क्रिप्ट कोड का प्रयोग करना आवश्यक हो तो "js" एक्सटेंशन वाली बाह्य फाइल में जावास्क्रिप्ट कोड को रखना सबसे अच्छा उपागम है। ऐसा करने के लिए, हम स्रोत जावास्क्रिप्ट फाइल से लिंक को दर्शाने के लिए **<SCRIPT>** टैग में Src विशेषता का प्रयोग करेंगे।

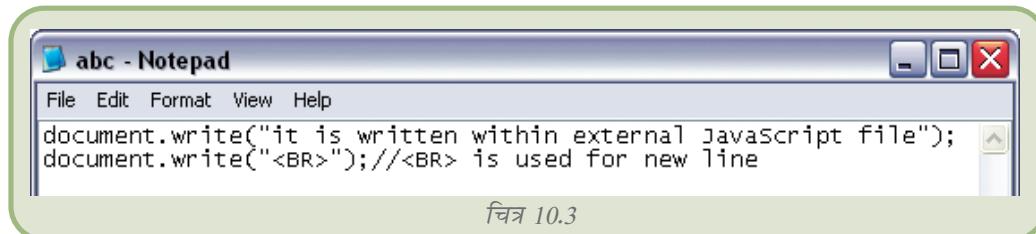
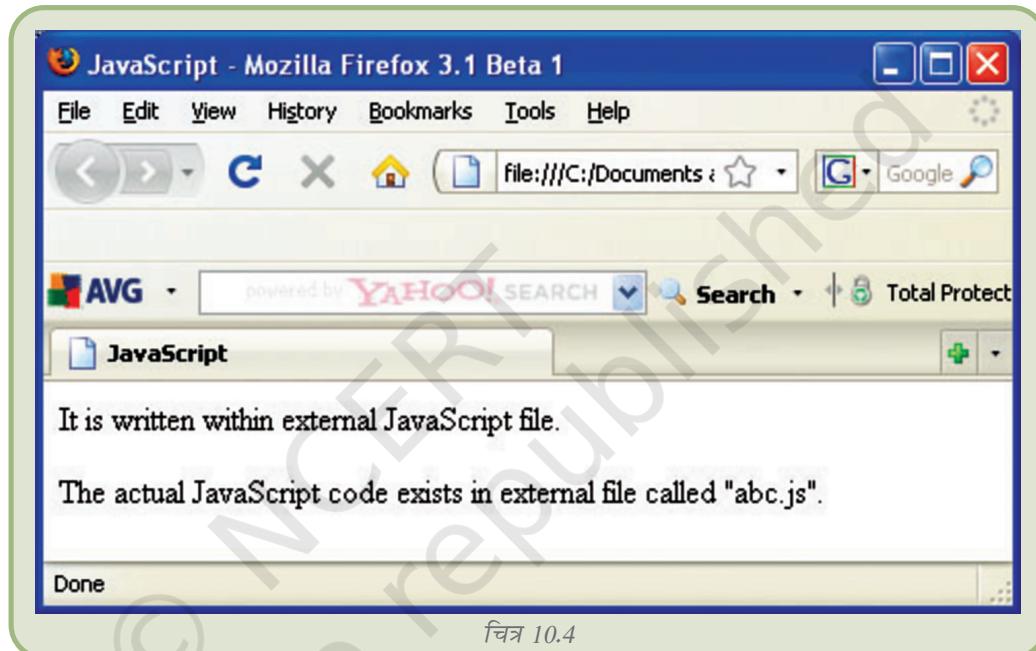
उदाहरण– बाह्य जावास्क्रिप्ट कोड के प्रयोग का उदाहरण।

```
<HTML>
<HEAD>
<TITLE>Using External JavaScript</TITLE>
</HEAD>
<BODY>
<SCRIPT language="JavaScript" src="abc.js">
</SCRIPT>
<P> The actual JavaScript code exists in external file called
"abc.js". </P>
</BODY>
</HTML>
```

वास्तविक जावास्क्रिप्ट फाइल “abc.js”

चित्र 10.3

परिणाम

चित्र 10.4

10.4 जावास्क्रिप्ट में कथन

कथन वे कमांड या निर्देश हैं जो कुछ निर्देशित कार्यों को करने के लिए जावास्क्रिप्ट इंटरप्रेटर को दिए जाते हैं। जावास्क्रिप्ट इंटरप्रेटर लगभग सभी इंटरनेट ब्राउज़रों में होता है। कार्य पूरा करने हेतु कथनों के समूह को स्क्रिप्ट या प्रोग्राम कहते हैं। जावास्क्रिप्ट कथन निम्नवत् होंगे–

```
a = 100;                    // stores value 100 in variable a
b = 200;                    // stores value 200 in variable b
c = a + b;                  // stores the sum of a and b in
                            // variable c
document.write
("Sum of A and B : ");      // displays the string
document.write(c);          // displays the value of c
```

जावास्क्रिप्ट में कथन की समाप्ति पर अर्धविराम (;) का प्रयोग किया जाता है लेकिन यदि अलग–अलग पंक्ति में दो कथन लिखे हों तो अर्धविराम हटाया जा सकता है। कुछ वैध कथन हैं–

(i) p = 10
q = 20
(ii) x = 12; y = 25 // semicolon (;) separating two statements.

कुछ अवैध कथन हैं–
x = 12; y = 25 // statements within the same line not seprated by semicolon (;)

10.4.1 कमेंट्स

कमेंट वह कथन है जिसकी इंटरप्रेटर द्वारा हमेशा अनदेखी की जाती है। कथन पर टिप्पणी देने के लिए इसका प्रयोग होता है जो इसे अन्य प्रोग्रामरों हेतु अधिक समझने योग्य और पठनीय बनाता है। कमेंट दो प्रकार के होते हैं–

- एकल पंक्ति कमेंट में डबल-स्लैश (//) का प्रयोग।
- बहु-पंक्ति कमेंट में /* एवं */ का प्रयोग।

उदाहरण के लिए –

// यह एकल-पंक्ति कमेंट है।
/* यह बहु-पंक्ति कमेंट है।
इसकी लंबाई कितनी भी हो सकती है। */

10.4.2 लिटरल्स

लिटरल्स उन स्थिर मान को निर्देशित करता है, जिसका जावास्क्रिप्ट कोड में प्रत्यक्ष रूप से प्रयोग होता है। उदाहरण के लिए–

a = 10;
b = 5.7;
documents.write ("welcome");
उपर्युक्त कथन में 10, 5.7, "welcome" लिटरल्स है।

10.4.3 आइडेन्टिफायर्स

आइडेन्टिफायर्स प्रोगामर द्वारा निर्मित चरों, कार्यो, विन्यास इत्यादि के नामों को निर्देशित करता है। यह संख्या सहित या अंडरस्कोर और डॉलर चिह्न सहित बड़े और छोटे शब्दों में अक्षरों का कोई भी क्रम हो सकता है। एक आइडेन्टिफायर संख्या के साथ आरंभ नहीं होना चाहिए और जावास्क्रिप्ट के किसी भी कीवर्ड के समान नाम का नहीं होना चाहिए।

कुछ वैध आइडेन्टिफायर्स हैं–
RollNo
bus_fee
_vp
$amt

कुछ अवैध (invalid) आइडेन्टिफायर्स हैं–

to day // जगह की अनुमति नहीं है।

17 nov // संख्या के साथ आरंभ नहीं होना चाहिए।

% age // किसी विशेष मद की अनुमति नहीं है।

10.4.4 आरक्षित शब्द या कीवर्ड्स

जावास्क्रिप्ट इंटरप्रेटर को निर्देश देने के लिए आरक्षित शब्दों का प्रयोग किया जाता है और प्रत्येक आरक्षित शब्द का एक विशिष्ट अर्थ होता है। प्रोग्राम में आइडेन्टिफायर्स के रूप में इनका प्रयोग नहीं किया जा सकता है। इसका अर्थ है, हम आरक्षित शब्दों को चरों, विन्यास, वस्तु कार्यों इत्यादि के नामों के रूप में प्रयोग नहीं कर सकते हैं। इन शब्दों को 'कीवर्ड' भी कहा जाता है। जावास्क्रिप्ट में आरक्षित शब्दों की सूची परिशिष्ट 10.1 में दी गई है।

10.4.5 चर

चर एक आइडेन्टिफायर है जो मान को सुरक्षित कर सकता है। इन मानों को स्क्रिप्ट के कार्यान्वयन के दौरान परिवर्तित किया जा सकता है। एक बार चर में जो मान सुरक्षित कर लिया जाता है तब उसे चर के नाम के प्रयोग से ही भेज सकते हैं। चर की घोषणा आवश्यक नहीं है, यद्यपि चर घोषणा के लिए यह एक अच्छा अभ्यास है, कीवर्ड **var** चर घोषणा के लिए प्रयोग होता है।

सिंटेक्स

var var-name [= value] [..., var-name [= value]]

उदाहरण

var name = "Sachin"; // यहाँ 'नाम' चर है।

documents.write (name); // सचिन प्रिंट होता है।

जावास्क्रिप्ट चर किसी भी प्रकार के डाटा के मान को होल्ड कर सकता है। उदाहरण के लिए–

i = 7;

documents.write(i); // 7 प्रिंट होता है।

i = "seven"; // जावास्क्रिप्ट में स्ट्रिंग मान रखा जा सकता है।

documents. write (i); // seven प्रिंट होता है।

चर घोषणा के कुछ वैध उदाहरण–

var cost;

var num, cust_no = 0;

var amount = 2000;

नामांकन पद्धति

हमें चर हेतु अर्थपूर्ण नाम का प्रयोग करना चाहिए। एक चर का नाम शब्द, अंडरस्कोर (_), या डॉलर चिन्ह ($) के साथ ही आरंभ करना चाहिए। इसके बाद के मद अंक (0–9) भी हो सकते हैं। जावास्क्रिप्ट स्थिति अनुसार कार्य करता है, इसलिए my_school और My_school इस चर का नाम समान नहीं है।

कुछ वैध चरों के नाम

f_name
India 123
_sumof

कुछ अवैध चरों के नाम

10_numbers – किसी भी संख्या से आरंभ नहीं होना चाहिए।
rate% – '%' एक वैध मद नहीं है।
my name – जगह छोड़ने की अनुमति नहीं है।

प्रोग्राम 10.2– var के प्रयोग से दो संख्याओं का योग निकालना।

```
<HTML>
<HEAD>
<TITLE>Sum of two numbers</TITLE>
</HEAD>
<BODY>
<SCRIPT type="text/javascript">
var a = 100;
var b = 500;
var c = a + b;
document.write ("Sum of a & b ↵
: "  + c );
</SCRIPT>
</BODY>
</HTML>
```

Sum of two numbers - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Sum of a & b : 600
Done

चित्र 10.5

10.5 डाटा के प्रकार

जावास्क्रिप्ट तीन मूल डाटा प्रकारों – संख्या, स्ट्रिंग, बूलीयन एवं दो कंपोजिट प्रकार के डाटा–विन्यास एवं वस्तु का समर्थन करता है।

10.5.1 संख्या

संख्या चर में किसी भी प्रकार की संख्या या तो पूर्ण संख्या या वास्तविक संख्या हो सकती है। संख्या के उदाहरण हैं–

29, –43, 3.40, 3.4323

10.5.2 स्ट्रिंग

स्ट्रिंग शब्दों, अंकों, विराम–चिह्नों इत्यादि का संग्रह है। स्ट्रिंग लिटरल को सिंगल कोट या डबल कोट (''or'') में रखा जाता है। स्ट्रिंग लिटरल के उदाहरण निम्न हैं–

'welcome', "7.86", "wouldn't you exit now", 'country = "India" '

जावास्क्रिप्ट में हम स्ट्रिंग लिटरल में एस्केप क्रम का प्रयोग कर सकते हैं। एस्केप क्रम बैकस्लैश (\) से आरंभ होता है, जिसके बाद अन्य मद आता है। यह बैकस्लैश मद का प्रतिनिधित्व करने या विशेष क्रिया प्रस्तुत करने के लिए ब्राउज़र को निर्दिष्ट करता है। उदाहरण के लिए, \" यह एक एस्केप क्रम है जो कि डबल कोट (") को दर्शाता है।

एस्केप क्रम	*प्रस्तुत एक्शन/मद*
\b	बैकस्पेस
\n	नयी पंक्ति
\r	कैरिज रिटर्न
\t	टैब
\'	सिंगल कोट (')
\"	डबल कोट ('')
\\	बैकस्लेश (\)

उदाहरण–

```
document.write ("Abhinav said, \"Earth doesn\'t revolve round ¿
the sun". But teacher corrected him.");
```

यहां इस उदाहरण में दो प्रकार के एस्केप क्रमों \" एवं \' का प्रयोग हुआ है।

आउटपुट

Abhinav said , "Earth doesn't revole round the sun". But teacher corrected him.

10.5.3 बूलियन मान

बूलियन चर केवल दो संभावित मान, सही या गलत, को सुरक्षित कर सकता है। आंतरिक रूप से इसे सही हेतु 1 और गलत हेतु 0 के रूप में सुरक्षित किया जाता है। इसका प्रयोग स्थितियों के परिणाम जानने के लिए होता है, चाहे स्थिति का परिणाम सही हो या गलत।

उदाहरण

x = = 100; // results true if x = 100 otherwise false.

10.5.4 एरेज (विन्यास)

विन्यास डाटा मान के समान प्रकारों का संग्रह है जिनका समान नाम होता है। विन्यास में प्रत्येक डाटा तत्व विन्यास में इसकी स्थिति द्वारा निर्देशित होता है जिसे सूची संख्या भी कहा जाता है। वैयक्तिक विन्यास तत्व को विन्यास के नाम द्वारा निर्देशित किया जा सकता है जो बाद में वर्ग कोष्ठक युग्म सूची संख्या सहित होता है। जावास्क्रिप्ट में सूची संख्या शून्य से आरंभ होती है अर्थात् जावास्क्रिप्ट में पहले तत्व में इसका सूची मान 0 होता है, द्वितीय सूची मान 1 होता है इत्यादि। विन्यास को निम्नलिखित रूप से किसी भी तरह लिखा जा सकता है–

```
var a = new a ( );
var x = [ ];
var m = [ 2, 4, "sun"];
```

एक विन्यास के विशिष्ट मान को आरंभ किया जाता है क्योंकि इसके विनिर्दिष्ट तर्कों की संख्या के अनुसार निर्धारित की जाती है।

उदाहरण– यह तीन तत्वों के साथ तीन खेलों के नाम के विन्यास को निर्मित करता है।

```
games = ["Hockey", "Cricket", "Football"];
```

हम विन्यास में विभिन्न प्रकार के मान सुरक्षित कर सकते हैं

उदाहरण हेतु–

```
var arr = new Array();        // विन्यास का निर्माण
arr[0] ="JAVASCIPT";          // सूची 0 में स्ट्रिंग लिटरल सुरक्षित करना
arr[1] = 49.5;                // सूची 1 में पूर्ण संख्या सुरक्षित करना
arr[2] = true;                // बूलियन मान सुरक्षित करना
```

10.5.5 नल मान

जावास्क्रिप्ट विशेष प्रकार के डाटा का समर्थन करता है जिसे **नल** कहते हैं जो ''रिक्त या मान न होना'' की ओर संकेत करता है। नोट करें कि '**नल**' शून्य के समान नहीं होता है।

उदाहरण

```
var distance = new object ();
distance = null;
```

10.6 वस्तुएं

जावास्क्रिप्ट वस्तु-आधारित स्क्रिप्टिंग भाषा होती है। इससे हम अपनी वस्तुएं परिभाषित कर सकते हैं और अपने चर बना सकते हैं। यह पूर्व परिभाषित वस्तु के सेट भी प्रस्तुत करता है। हमारे वेब पेज पर सारणियां, फॉर्म, बटन, इमेज या लिंक्स सभी वस्तुओं का उदाहरण है। वस्तु से संबंधित मान उसकी विशेषता है और वस्तु द्वारा जो एक्शन किए या प्रस्तुत किए जाते हैं वह विधि या व्यवहार है। वस्तु संबंधित विशेषता निम्नवत् रूप से प्राप्त होती है–

```
ObjectName.Property Name
```

अब हम जावास्क्रिप्ट में कुछ पूर्व परिभाषित वस्तुओं के बारे में अध्ययन करेंगे।

10.6.1 प्रलेख वस्तु

प्रलेख वस्तु विंडो वस्तु का एक भाग होता है। इस तक **window.document** विशेषता द्वारा पहुँचा जा सकता है। प्रलेख वस्तु HTML प्रलेख को प्रस्तुत करता है और HTML प्रलेख में सभी तत्वों तक पहुँचा जा सकता है। उदाहरण के लिए; **document.title** विशेषता द्वारा वर्तमान प्रलेख के शीर्षक तक पहुँच सकते हैं।

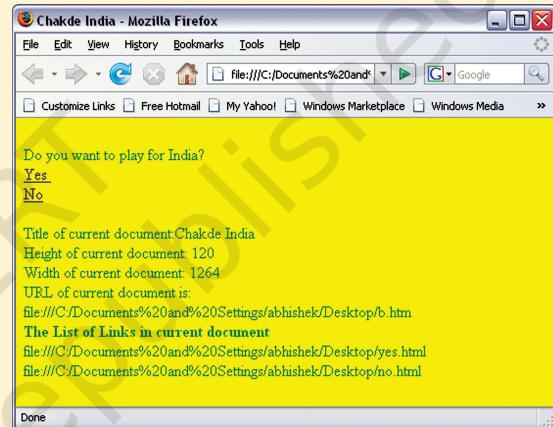
प्रलेख वस्तु की कुछ सामान्य विशेषताएँ हैं–

प्रोपर्टीज	*उद्देश्य*
Tittle	वर्तमान प्रलेख के टाइटल को दर्शाना / सेट करना।
bgcolor	वर्तमान प्रलेख की पृष्ठभूमि का रंग दर्शाना / सेट करना।
fgcolor	वर्तमान प्रलेख के टेक्स्ट का रंग दर्शाना / सेट करना।
linkcolor	प्रलेख में हायपरलिंक का रंग दर्शाना / सेट करना।
alinkcolor	प्रलेख में चालू लिंक्स का रंग दर्शाना / सेट करना।
vlinkcolor	देखे गए हायपरलिंक्स का रंग दर्शाना / सेट करना।
height	वर्तमान प्रलेख की ऊंचाई दर्शाना।
width	वर्तमान प्रलेख की चौड़ाई दर्शाना।
Forms	वर्तमान प्रलेख के अंतर्गत फॉर्म तत्वों की सूची दर्शाना।
Images	वर्तमान प्रलेख की इमेजों की सूची दर्शाना।
URL	वर्तमान प्रलेख के यूआरएल युक्त स्ट्रिंग दर्शाना।
Location	वर्तमान प्रलेख विंडो में अन्य यूआरएल लोड करना।

विधि	***उद्देश्य***
Open ()	लिखने हेतु प्रलेख खोलता है।
write ()	प्रलेख में स्ट्रिंग / डाटा लिखता है।
writeln ()	स्ट्रिंग / डाटा और तत्पश्चात् प्रलेख में नई पंक्ति के करेक्टर को लिखता है।
close ()	लिखने हेतु प्रलेख स्ट्रीम बंद करना।

प्रोग्राम 10.3– प्रलेख वस्तु की विशेषताओं और विधियों की व्याख्या करना।

```
<HTML>
<HEAD>
<TITLE>Document Properties</TITLE>
</HEAD>
<BODY>
<SCRIPT type=”text/javascript”>
```

```
document.fgColor = “green”; // sets text color
document.bgColor = “yellow”;      // background color
document.title = “Chakde India”;  // change title
document.linkColor = “navy”;      // hyperlinks color
document.alinkColor = “red”;      // active links
document.vlinkColor = “lime”;     // visited hyperlinks
document.write(“<BR>Do you want to play for India?”);
document.write(“<BR> <A href=’yes.html’> Yes </A>”);
document.writeln(“<BR> <A href=’no.html’>No</A><BR>”);
document.write(“<BR>Title of current document: “ + document.title);
document.write(“<BR>Height of current document: “ + document.height);
document.write(“<BR>Width of current document: “ + document.width);
document.write(“<BR> URL of current document is: “ + document.URL);
//Use of document.links to list of all the hyperlinks
document.write(“<BR><B>The List of Links in current document</B>”);
var links = document.links;
for(var i = 0; i < links.length; i++)
{
document.write(“<BR>”+ document.links[i]);
}
</SCRIPT>
</BODY>
</HTML>
```

चित्र 10.6

10.6.2 दिनांक संबंधी वस्तु

इस वस्तु का प्रयोग समय एवं दिनांक को सेट करने तथा सही करने के लिए किया जाता है। जावास्क्रिप्ट तिथियां 1 जनवरी 1970 की अर्धरात्रि से मिलीसैकेंड की संख्या के रूप में सुरक्षित होती हैं। इस दिनांक को ईपॉक कहा जाता है। 1970 से पहले के दिनांक को ऋणात्मक संख्या द्वारा प्रस्तुत किया जाता है। दिनाँक ( ) सहित नये कीवर्ड के प्रयोग द्वारा दिनांक वस्तु को निर्मित किया जाता है।

सिंटेक्स

```
new Date ()
new Date (milliseconds)
new Date (dateString)
new Date (yr_num, mo _ num, day _ num
[,hr_num, min_num, sec_num, ms_num])
```

पैरामीटर

Milliseconds	1 जनवरी 1970 से मिलीसैकेंड 00:00:00.
dateString	डेट स्ट्रिंग उदाहरण. ''अक्तूबर 5, 2007''
yr_num, mo_num, day_num	वर्ष (उदाहरण. 2007) माह (मान 0-11, जनवरी के लिए 0 और दिसम्बर के लिए 11), दिन (1-31)
hr_num, min_num, sec_num, ms_num	घंटा, मिनट, सैकेंड एवं मिलीसैकेंड का मान

Date() प्रयोग करने के विभिन्न उदाहरण

```
today = new Date();
dob = new Date("October 5, 2007 12:50:00");
doj = new Date(2007,10,5);
bday = new Date(2007,10,5,12,50,0);
```

दिनाक मान पठन की विधियां

दिनाक वस्तु से मान प्राप्त करने के लिए हम गेट (**get**) विधि का प्रयोग कर सकते हैं। यहां कुछ गेट विधियां हैं जो स्थानीय समय के अनुसार कुछ मान प्रदान करती हैं।

getDate ()	माह के दिन को दर्शाना
getDay ()	सप्ताह के दिन को दर्शाना
getFullyear ()	पूर्ण वर्ष को दर्शाना
getHours ()	घंटे को दर्शाना
getMinutes ()	मिनट को दर्शाना
getMonth ()	माह को दर्शाना
getSeconds ()	सैकेंड को दर्शाना
getTime ()	समय के अनुरूप सांख्यिकी मान दर्शाना
getyear ()	वर्ष को दर्शाना

प्रोग्राम 10.4– सामान्य जावास्क्रिप्ट प्रोग्राम जो आज की दिनांक और वर्तमान समय डिसप्ले करता है।

```
<HTML>
<HEAD>
    <TITLE>Displaying Time</TITLE>
</HEAD>
<BODY>
    <CENTER>
          <H1>Today's Date and Current Time</H1>
    </CENTER>
    <SCRIPT type="text/javascript">
          var today = new Date();
          document.write("<H2>"); // JavaScript allows the use
          document.write(today);  // of HTML formatting tag
```

```
            document.write("</H2>"); // with document.write
      </SCRIPT>
   </BODY>
   </HTML>
```

चित्र 10.7

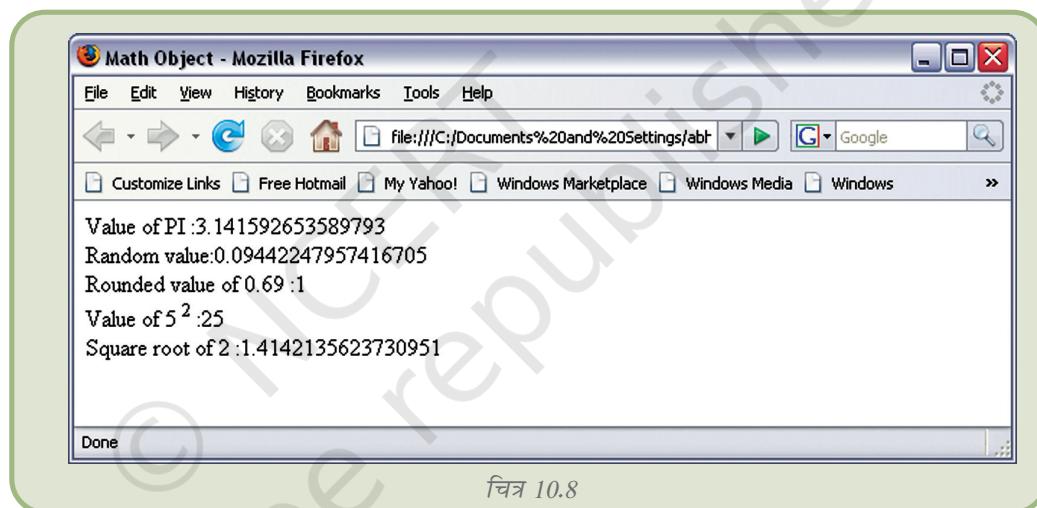
10.6.3 गणित (Math) वस्तु

इस वस्तु में विधियां और स्थिर मान होते हैं जो जटिल गणितीय प्रश्नों के लिए होते हैं। यह अन्य वस्तुओं के समान तत्काल आरंभ नहीं किया जा सकता। ***गणित*** की सभी विशेषताएं और विधियां स्थिर हैं। हम स्थिर p को ***Math.PI*** और साइन कार्य को ***Math.sin (x)*** के रूप में निर्देशित कर सकते हैं, जहाँ ***x*** विधि तर्क है।

विशेषताएँ	*विवरण*
Math.PI	p के मान को दर्शाना।
Math.E	यूलर स्थिरांक तथा प्राकृतिक लघुगणकों का आधार।
Math.LN2	2 का प्राकृतिक लघुगणक।
Math.LN10	10 का प्राकृतिक लघुगणक, लगभग 2.302।
SQRT1_2	½ का वर्गमूल।
SQRT2	2 का वर्गमूल।
विधियाँ	*विवरण*
Pow(x, p)	x^p दर्शाना।
abs(x)	x का निरपेक्ष मान दर्शाना।
exp(x)	e^x दर्शाना।
log(x)	x का प्राकृतिक लघुगणक दर्शाना।
sqrt(x)	x का वर्गमूल दर्शाना।
random()	0 और 1 में मध्य की यादृच्छिक संख्या को दर्शाना।
ceil(x)	x से बड़ी या समान छोटी पूर्ण संख्या को दर्शाना।
floor(x)	x से छोटी या समान बड़ी पूर्ण संख्या को दर्शाना।
min(x, y)	x और y में कम को दर्शाना।
max(x, y)	x और y में अधिक को दर्शाना।
round(x)	निकट की पूर्ण संख्या से x के ऊपर या नीचे आना।
sin(x)	x sin पद दर्शाना, जहां x रेडियन में हो।
cos(x)	cosin x दर्शाना, जहां x रेडियन में हो।
tan(x)	tan x दर्शाना, जहां x रेडियन में हो।

उदाहरण– गणित वस्तु की विशेषताओं और विधियों की व्याख्या करना।

```
<HTML>
<HEAD>
<TITLE>Math Object</TITLE>
</HEAD>
<BODY>
<SCRIPT type="text/JavaScript">
document.write("Value of PI :"+Math.PI + "<BR>");
document.write("Random value:"+Math.random()+"<BR>");
document.write("Rounded value of 0.69 :"+ ↵
Math.round(0.69)+"<br>");
document.write("Value of 5 <SUP>2</SUP> :"+ ↵
Math.pow(5,2) + "<br>");
document.write("Square root of 2 :"+Math.SQRT2 );
</SCRIPT>
</BODY>
</HTML>
```

चित्र 10.8

10.7 अभिव्यक्तियां एवं ऑपरेटर्स

अभिव्यक्ति ऑपरेटर्स और ऑपरैंड का मिश्रण है जिसका मूल्यांकन किया जा सकता है। इसमें फंक्शन कॉलम भी हैं जो मान प्रदान करते हैं–

उदाहरण

`x = 7.5`	// सांख्यिकी लिटरल है।
`"Hello India!"`	// स्ट्रिंग लिटरल है।
`false`	// बूलियन लिटरल है।
`{feet:10, inches:5}`	// वस्तु लिटरल है।
`[2,5,6,3,5,7]`	// विन्यास लिटरल है।
`v= m + n;`	// v चर लिटरल है।
`tot`	// tot चर लिटरल है।

10.7.1 अंकगणितीय ऑपरेटर्स

इसका अंकगणितीय/गणितीय कार्यों जैसे– घटा, भाग, गुणा इत्यादि करने के लिए प्रयोग किया जाता है। अंकगणितीय ऑपरेटर एक या अधिक अंकीय मान (या तो लिटरल या चर) पर कार्य करता है और एकल अंकीय मान प्रदान करता है। मुख्य अंकगणितीय ऑपरेटर हैं–

+ (जमा)   - (घटा)

* (गुणा)   / (भाग)

% (मॉड्यूल्स)   ++(1 द्वारा इंक्रीमेंट)

– (1 द्वारा डिक्रीमेंट)

उदाहरण

```
var s = 10 + 20;          // परिणामः s=30
var h = 50 * 4;           // परिणामः h = 200
var d = 100 / 4;          // परिणामः d = 25
var r = 72 % 14;          // परिणामः r=2
```

इंक्रीमेंट और डिक्रीमेंट ऑपरेटर्स

इन ऑपरेटर्स का प्रयोग चर का मान एक अंक बढ़ाने या घटाने हेतु किया जाता है। इन ऑपरेटर्स के प्रयोग से गणना तेजी से होती है।

उदाहरण

```
var a = 15;
a++;                      // परिणामः a = 16
var b = 20;
b--;                      // परिणामः b = 19
```

10.7.2 एसाइनमेंट ऑपरेटर्स

यह इसके दाएं ओपरैंड से बाएं ओपरैंड को मान प्रदान करता है। इस ऑपरेटर को बराबर के चिह्न (=) द्वारा प्रस्तुत किया जाता है।

उदाहरण

```
x = 100;                  // इस कथन में 100 से x तक मान हैं।
```

जावास्क्रिप्ट में मानक प्रचालनों के लिए शॉर्टहैंड ऑपरेटर भी है।

शार्टहैंड ऑपरेटर	उदाहरण	के बराबर
+=	a += b	a = a + b
-=	a -= b	a = a - b
*=	a *= b	a = a * b
/=	a /= b	a = a / b
%=	a %= b	a = a % b

10.7.3 संबंधात्मक (तुलनात्मक) ऑपरेटर

संबंधात्मक ऑपरेटर संकेत चिह्न हैं जो स्थिति का मूल्यांकन करने के पश्चात् सही या गलत बूलियन मान प्रदान करते हैं। उदाहरण के लिए $x > y$; सत्य मान प्रदान करता है जो सही है कि चर x का मान चर y से बड़ा है।

प्रमुख जावास्क्रिप्ट तुलनात्मक ऑपरेटर निम्नवत् सारणी में दिये गए हैं–

ऑपरेटर	*विवरण*	*उदाहरण*
= =	के बराबर	4 = = 8 गलत प्रदान करता है।
! =	के बराबर नहीं है	4 ! = 8 सत्य प्रदान करता है।
>	से बड़ा है	8 > 4 सत्य प्रदान करता है।
<	से छोटा है	8 < 4 गलत प्रदान करता है।
< =	से कम या बराबर है	8 < = 4 गलत प्रदान करता है।
> =	से बड़ा या बराबर है	8 > = 4 सत्य प्रदान करता है।

संबंधात्मक ऑपरेटर स्ट्रिंग हेतु भी कार्यात्मक है। तुलना वर्णक्रमानुसार होती है। यह वर्णक्रम ASCII संख्या पर आधारित होता है। उदाहरण के लिए–

कथन	***आउटपुट***
`“zero” < “one”`	// गलत
`“Zero” < “one”`	// सही
`10 < 5`	// गलत, अंकीय तुलना।
`“10” < “5”`	// सही, स्ट्रिंग तुलना।
`“10” < 5`	// गलत, अंकीय तुलना।
`“Ten” < 5`	// “Ten” को संख्या में नहीं बदला जा सकता है।

10.7.4 तार्किक ऑपरेटर

तार्किक ऑपरेटर का प्रयोग दो या अधिक स्थितियों का मेल करने के लिए होता है। जावास्क्रिप्ट के तीन तार्किक ऑपरेटर हैं –

ऑपरेटर	*उदाहरण सहित विवरण*
&& (AND)	यदि दोनों ऑपरैंड सत्य हैं तो सत्य दर्शाता है नहीं तो गलत दर्शाता है।
|| (OR)	यदि दोनों ऑपरैंड गलत हैं तो गलत दर्शाता है नहीं तो सही दर्शाता है।
! (NOT)	यदि ऑपरैंड गलत है तो सही दर्शाता है और यदि ऑपरैंड सही है तो गलत दर्शाता है।

10.7.5 शृंखलात्मक ऑपरेटर

+ ऑपरेटर दो स्ट्रिंग ऑपरैंड को शृंखलाबद्ध करता है। + ऑपरेटर सांख्यिकी ऑपरैंड से अधिक

स्ट्रिंग ऑपरैंड को महत्त्व देता है, यह बाएं से दाएं कार्य करता है। परिणाम उस क्रम पर आधारित होता है, जिसमें कार्य होते हैं। उदाहरण के लिए–

कथन	*आउटपुट*
"Good" + "Morning"	"GoodMorning"
"5" + "10"	"510"
"Lucky" + 7	"Lucky7"
4 + 7 + "Delhi"	"11Delhi"
"Mumbai" + 0 + 0 + 7	"Mumbai007"

10.7.6 विशिष्ट ऑपरेटर

कंडीशनल ऑपरेटर (?:)

कंडीशनल ऑपरेटर एक विशेष जावास्क्रिप्ट ऑपरेटर है जिसमें तीन ऑपरैंड हैं। अत:, इसे टरनरी ऑपरेटर भी कहते हैं। कंडीशनल ऑपरेटर स्थिति के आधार पर चर का मान निर्धारित करता है।

सिंटेक्स

```
var_name = (condition) ? v_1 : v_2
```

यदि (कंडीशन) सही है, तो चर को v_1 मान दिया जाता है अन्यथा, चर को v_2 मान दिया जाता है।

उदाहरणार्थ

```
status = (age >= 18) ? "adult" : "minor"
```

इस कथन में चर स्थिति को 'एडल्ट' मान दिया गया है यदि आयु 18 अथवा इससे अधिक हो। अन्यथा यह स्टेटस को 'माइनर' मान देता है।

न्यू

'न्यू' ऑपरेटर का प्रयोग दृष्टांत सृजित करने और प्रयोक्ता द्वारा निर्धारित अथवा पूर्व–निर्धारित वस्तु प्रकारों को मैमोरी आबंटित करने के लिए किया जाता है।

सिंटेक्स

ObjectName = new objectType (param1 [,param2] ...[,paramN])

उदाहरण

```
d = new Date(); // date assigns to object d
r = new rectangle(4, 5, 7, 8);
```

डिलीट (लोप करना)

'डिलीट' ऑपरेटर मैमोरी स्पेस को निरआबंटित (निर्मुक्त) करता है जिसे किसी वस्तु की संपत्ति अथवा विन्यास से किसी तत्व को डिलीट कर नए ऑपरेटर के प्रयोग द्वारा आबंटित किया गया था।

इसका सिंटेक्स निम्न प्रकार है–

```
delete object_name
delete object_name.property
delete array_name[index]
```

delete ऑपरेटर का प्रयोग उन चरों का लोप करने के लिए किया जा सकता है जिनकी घोषणा बिना किसी संदेह के की गई है न कि उनके लिए जिन्हें अंत: कथन के साथ घोषित किया गया है। डिलीट ऑपरेटर 'सत्य' दर्शाता है यदि प्रचालन संभव है और असत्य दर्शाता है, यदि प्रचालन संभव नहीं है।

```
a=12
var j= 63
myobj=new Number()
myobj.h=55                 // create property h
delete x                   /* returns true (x is declared implicitly,
                           without using var)*/
delete y                   /* returns false ( y is declared
                           explicitly using var) */
delete Math.PI             /* returns false (cannot delete
                           predefined properties)*/
delete myobj.h             /* returns true (can delete user-defined
                           properties)*/
delete myobj               /* returns true (can delete if declared
                           implicitly) */
```

जब हम विन्यास में से किसी तत्व को हटाते हैं तो विन्यास की लंबाई प्रभावित नहीं होगी। जैसे– यदि हम a[3] का लोप करते हैं तो a[4] अपने रूप में विद्यमान रहेगा और a[3] परिभाषित नहीं होगा। जब डिलीट ऑपरेटर विन्यास तत्व को हटाता है तो वह तत्व विन्यास में विद्यमान नहीं रहता।

दिस (This)

जावास्क्रिप्ट **'दिस'** ऑपरेटर को समर्थन देती है। **'दिस'** शब्द से अभिप्राय वर्तमान वस्तु से है। यह वर्तमान वस्तु का प्वाइंटर जैसा है।

इसका सिंटेक्स है–

```
this[.propertyName]
```

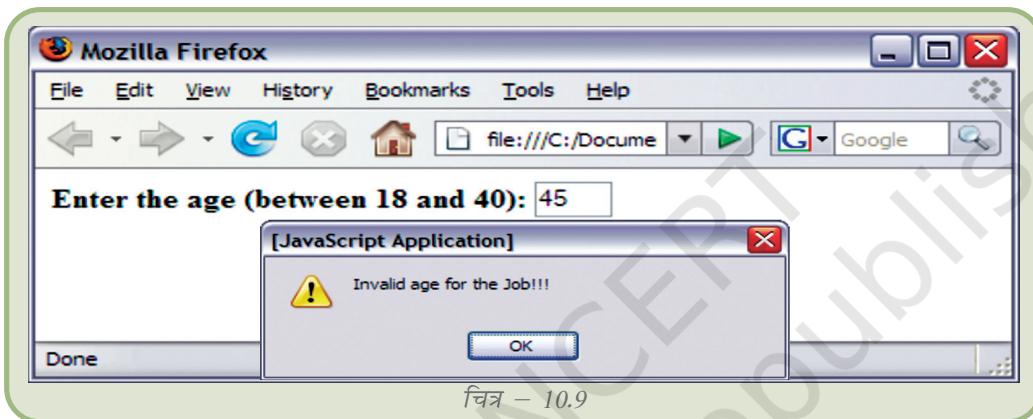
उदाहरण

दिस ऑपरेटर का प्रयोग आयु की वैधता के लिए किया जाता है। इसमें इनपुट टेक्स्ट बॉक्स के माध्यम से उपलब्ध कराई जाती है।

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
    function validate(obj, min_age, max_age)
{
   if ((obj.value < min_age) || (obj.value > max_age))
```

```
        alert("Invalid age for the Job!!!");
}
</script>
</HEAD>
<BODY>
<B>Enter the age (between 18 and 40):</B>
<INPUT TYPE = "text" NAME = "age" SIZE = 2
onChange="validate(this, 18, 40)">
</BODY>
</HTML>
```

इस उदाहरण में हमने ऑनचेंज इवेंट हैंडलर द्वारा वैलिडेट () फंक्शन आरंभ किया है। यहां पर दिस ऑपरेटर का प्रयोग वर्तमान वस्तु (अर्थात् टेक्स्ट बॉक्स) को पास करने हेतु किया जाता है।

चित्र – 10.9

10.7.7 ऑपरेटर प्राथमिकता

ऑपरेटरों का प्राथमिकता के पूर्व-निर्धारित क्रम में मूल्यांकन किया जाता है। निम्न सारणी में ऑपरेटरों की उच्चतम प्राथमिकता से निम्नतम प्राथमिकता का क्रम दर्शाया गया है–

सारणी 10.1: ऑपरेटर प्राथमिकता

ऑपरेटर	विवरण	प्राथमिकता
*	गुणा	उच्चतम ↑
/	भाग	
%	मॉड्यूल्स	
+	जमा	
–	घटा	
<	से कम	
< =	से कम बराबर	
>	से अधिक	
> =	से अधिक बराबर	

= = ! =	समान असमान	
& &	लॉजिकल AND	
| |	लॉजिकल OR (या)	
?:	कंडीशनल	
= + = – = * = / = % =	असाइनमेंट ऑपरेटर्स	
,	कॉमा	निम्नतम

10.8 जावास्क्रिप्ट पॉप-अप बॉक्सेज़ (डायलॉग बॉक्सेज़)

जावास्क्रिप्ट में तीन प्रकार के पॉप-अप बॉक्सेज़– अलर्ट बॉक्स, कन्फर्म बॉक्स और प्रॉम्प्ट बॉक्स को विन्डो वस्तु की तीन विधियों द्वारा सृजित किया जा सकता है।

10.8.1 अलर्ट बॉक्स

विंडो वस्तु की अलर्ट ( ) विधि लघु टेक्स्ट मैसेज और ''ओके'' कमांड बटन से एक छोटा डायलॉग बॉक्स सृजित करती है जिसे अलर्ट बॉक्स कहते हैं। अलर्ट बॉक्स में चेतावनी दर्शाने वाला आइकन होता है।

सिंटेक्स

```
[window].alert("Text to be displayed on the popup box");
```

'विंडो' शब्द वैकल्पिक है।

उदाहरण

```
window.alert("I am to alert you about ….");
or
alert("I am to alert you about ….");
```

आउटपुट

यदि हम प्रयोक्ता के लिए कोई सूचना प्रदर्शित करना चाहते हैं तो अलर्ट बॉक्स का प्रयोग किया जाता है। जब अलर्ट बॉक्स आता है तो प्रयोक्ता को आगे बढ़ने के लिए ''ओके'' बटन क्लिक करना होता है।

10.8.2 कन्फर्म बॉक्स

यदि हम चाहते हैं कि प्रयोक्ता द्वारा सूचना का सत्यापन और पुष्टि की जाए तो कन्फर्म बॉक्स का प्रयोग किया जाता है। प्रयोक्ता को या तो ''ओके'' या 'कैंसल' बटन पर क्लिक करना होगा।

सिंटेक्स

```
[window.]confirm("Text to be confirmed");
```

उदाहरण

```
confirm("Do you want to quit now?");
```

आउटपुट

कन्फर्म बॉक्स बूलियन मान प्रदान करता है। यदि प्रयोक्ता ''ओके'' बटन पर क्लिक करता है तो यह 'सही' दर्शाता है और यदि प्रयोक्ता 'कैंसल' पर क्लिक करता है तो यह 'गलत' दर्शाता है।

10.8.3 प्रॉम्प्ट बॉक्स

प्रॉम्प्ट बॉक्स से प्रयोक्ता इनपुट प्राप्त करने की अनुमति प्रदान करता है। हम डाले गए टेक्स्ट हेतु डिफॉल्ट टेक्स्ट विनिर्दिष्ट कर सकते हैं। प्रॉम्प्ट ( ) से प्रयोक्ता द्वारा भेजी गई जानकारी को चर में सुरक्षित किया जा सकता है।

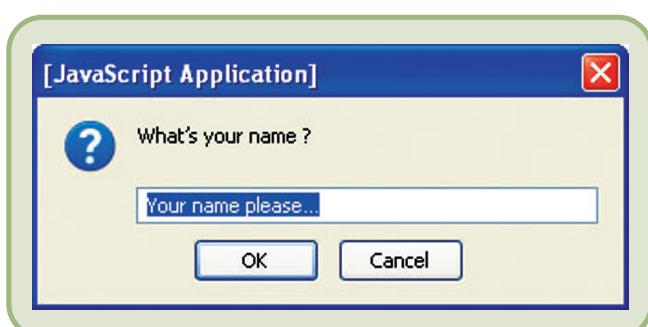
सिंटेक्स

```
prompt("Message" [, "default value in the text field"]);
```

उदाहरण

```
var name = prompt("What's your ↲
name? ", "Your name please…");
```

आउटपुट

प्रॉम्प्ट बॉक्स प्रयोक्ता द्वारा ''ओके'' पर क्लिक करने पर इनपुट स्ट्रिंग वैल्यू प्रदान करता है। यदि प्रयोक्ता 'कैंसल' पर क्लिक करता है तो यह 'नल' मान प्रदान करता है।

10.9 ब्लॉक कथन

दो या अधिक कथनों को जोड़कर कोष्ठकों के प्रयोग से एक ब्लॉक कथन बनाया जा सकता है।

इसका सिंटेक्स है

```
{
    statement_1
    statement_2
    .
    .
    statement_n
}
e.g.  if  (z > y)
{    x =10;
y = 20; }
```

10.10 ब्रांचिंग और लूपिंग कथन

जावास्क्रिप्ट द्वारा वैकल्पिक तरीकों में से चयन अथवा कथन या कथनों के ब्लॉक के कार्यान्वयन को दोहराया जा सकता है। जावास्क्रिप्ट ब्रांचिंग हेतु कुछ कंडीशनल कथनों का समर्थन करती है। कंडीशनल कथन वह कथन है जिसका प्रयोग हम कंडीशन आधारित कोड के भाग को कार्यान्वित करने अथवा कंडीशन पूरी न होने की दशा में कुछ अन्य करने के लिए कर सकते हैं।

लूपिंग, कथनों के सेट के कार्यान्वयन को कई बार दोहराना है।

10.10.1 ब्रांचिंग (कंडीशनल) कथन

यदि (if) कथनों के साथ ब्रांचिंग

तार्किक अभिव्यक्ति (कंडीशन) के आधार पर किसी कथन या कथनों के ब्लॉक का कार्यान्वयन करने हेतु 'इफ' कथन का प्रयोग किया जाता है। इसके तीन प्रकार हैं–

–इफ... कथन (साधारण ***'इफ'*** कथन)

–इफ... एल्स कथन

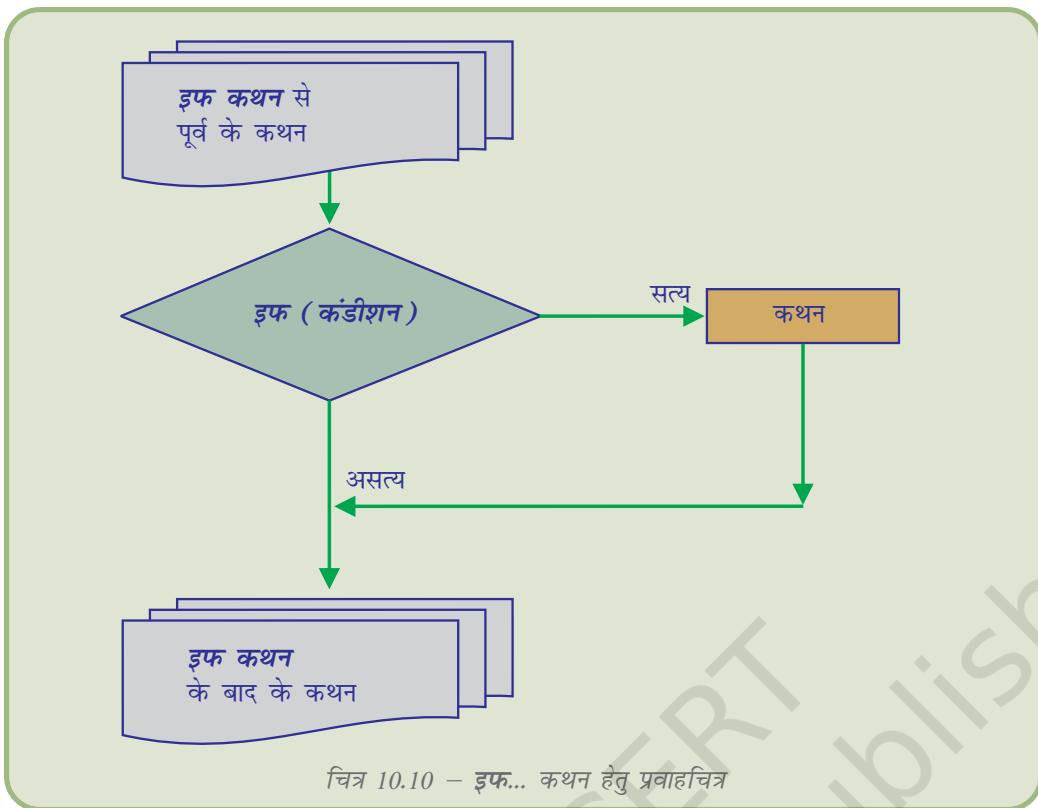
–इफ... एल्स इफ... कथन (***एल्स इफ*** क्रमिक कथन)

इफ...कथन

'इफ' (यदि) एक सामान्य निर्णयन कथन है। इसका प्रयोग कथनों के लिए तभी किया जाता है यदि विनिर्दिष्ट कंडीशन सत्य हो।

इसका सिंटेक्स है–

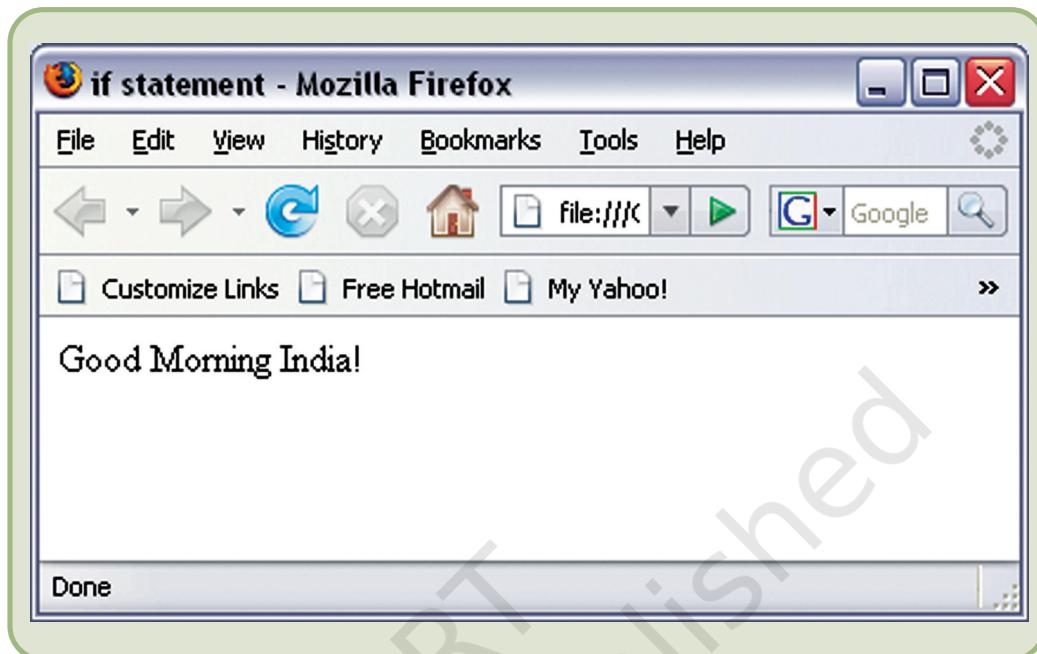
```
if ( condition )
{
.. statement(s) to be executed if (condition)  is true...
}
```

चित्र 10.10 – इफ... कथन हेतु प्रवाहचित्र

ऊपर के प्रवाहचित्र में कथन तभी कार्यान्वित किए जाएंगे जब तार्किक अभिव्यक्ति सही हो। अन्यथा, 'इफ कथन' के बाद के कथन को तत्पश्चात् कार्यान्वित किया जाएगा।

उदाहरण: 'गुडमॉर्निंग इंडिया!' डिस्प्ले करने वाला जावास्क्रिप्ट प्रोग्राम केवल तभी पेज पर आएगा यदि वेब पेज पर समय 12 घंटे से कम हो अन्यथा पेज खाली रहेगा।

```
<HTML>
<HEAD>
    <TITLE>if statement</TITLE>
</HEAD>
<BODY>
<script type = "text/javascript">
var d = new Date();
var time = d.getHours();  // time stores hours
if (time < 12)
{
    document.write("Good Morning India!");
}
</script>
</BODY>
</HTML>
```

यह एचटीएमएल प्रलेख ''गुड मॉर्निंग इंडिया!'' मैसेज को प्रदर्शित करेगा यदि आपके सिस्टम का समय 12 घंटे से कम है; अन्यथा पेज खाली दिखेगा।

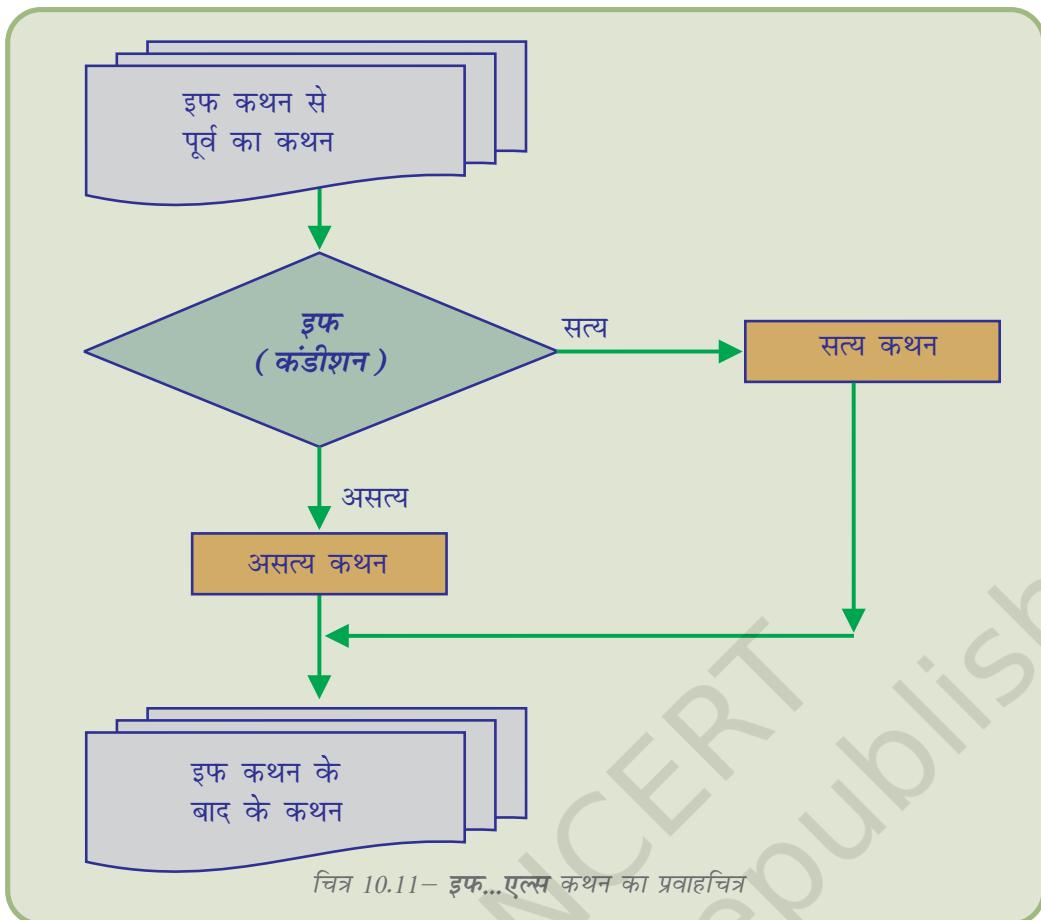
आउटपुट

इफ...एल्स कथन (if...else statement)

यह कथन सामान्य 'इफ' कथन का विस्तार है। इससे दो कथन या कथनों के समूह में से तार्किक जांच के आधार पर एक को लिया जा सकता है।

इसका सिंटेक्स है

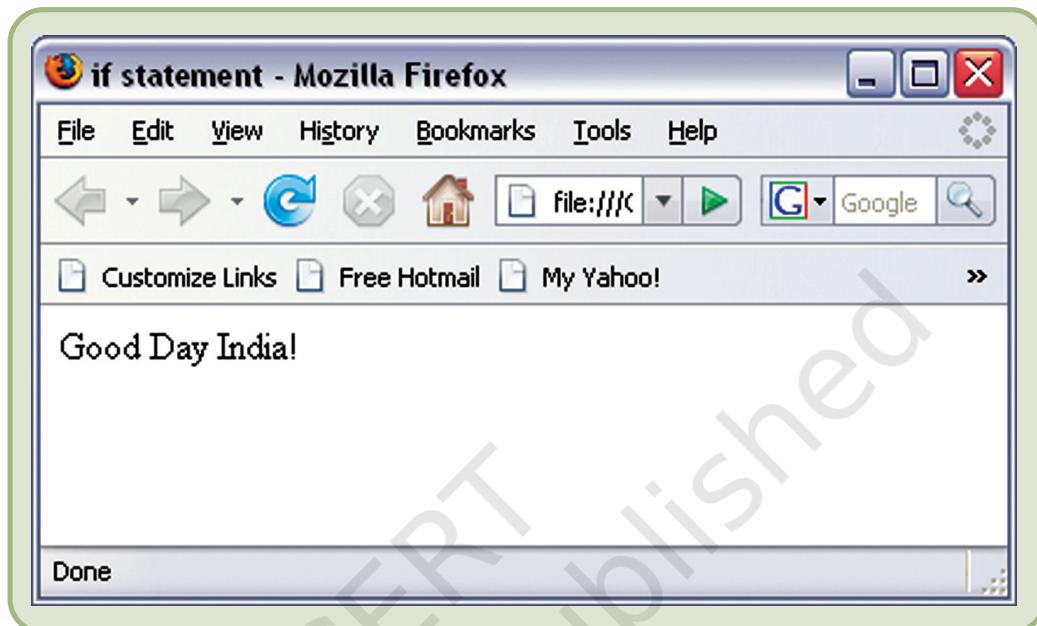
```
if ( condition )
{
    True statement(s)...
}
else
{
    False statement(s)...
}
```

यदि तार्किक अभिव्यक्ति (कंडीशन) सही है तो सही कथन कार्यान्वित होगा अन्यथा असत्य कथन कार्यान्वित किया जाएगा।

*चित्र 10.11– **इफ...एल्स** कथन का प्रवाहचित्र*

उदाहरण– जावास्क्रिप्ट प्रोग्राम जो यदि समय 12 घंटे से कम हो तो, 'गुड मॉर्निंग इंडिया!' प्रदर्शित करता है अन्यथा यह प्रलेख में 'गुड डे इंडिया!' दर्शाता है।

```
<HTML>
<HEAD>
    <TITLE>if else statement</TITLE>
</HEAD>
<BODY>
<script type = "text/JavaScript">
var d = new Date();
var time = d.getHours();
if (time < 12)
{
    document.write("Good Morning India!");
}
else
{
    document.write("Good Day India!");
}
</SCRIPT>
</BODY>
</HTML>
```

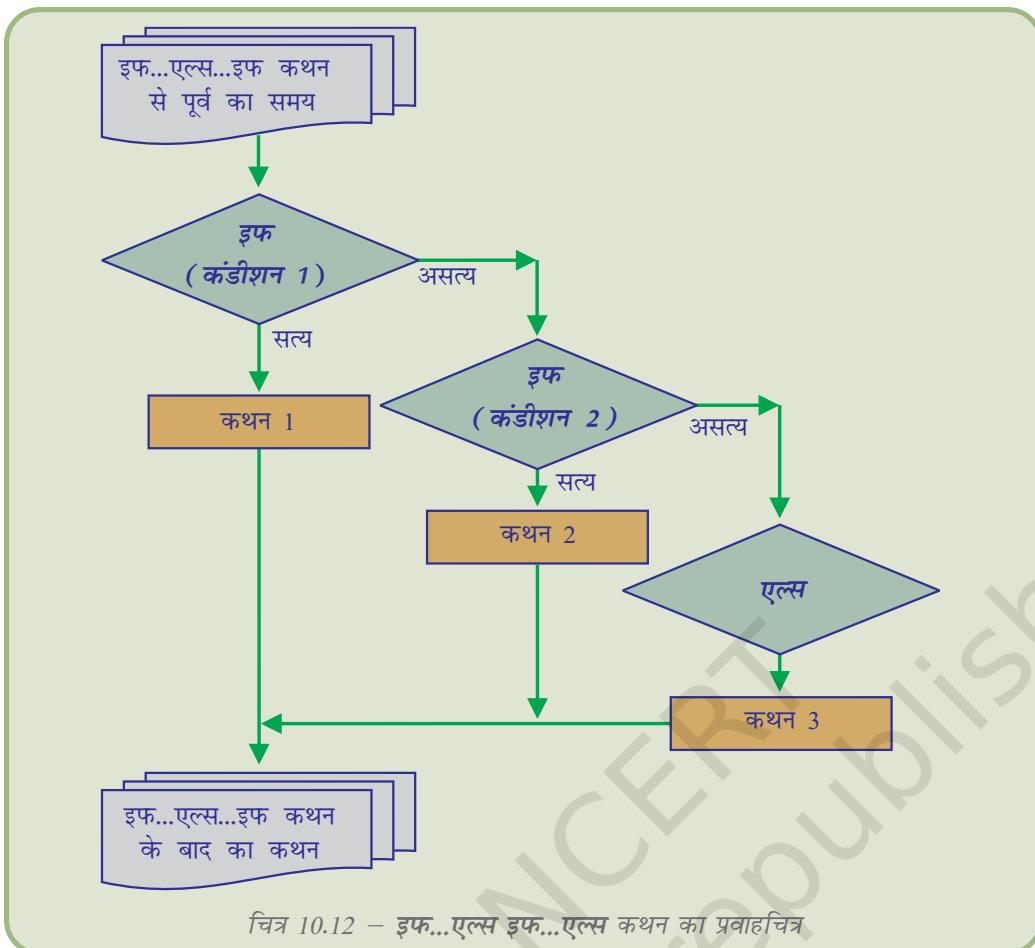
एचटीएमएल प्रलेख 'गुड मॉर्निंग इंडिया!' मैसेज दर्शाता है यदि सिस्टम का समय 12 घंटे से कम हो अन्यथा यह 'गुड डे इंडिया!' मैसेज दर्शाता है।

आउटपुट

इफ...एल्स इफ...एल्स कथन

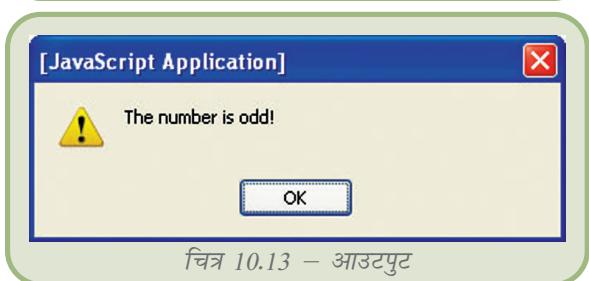
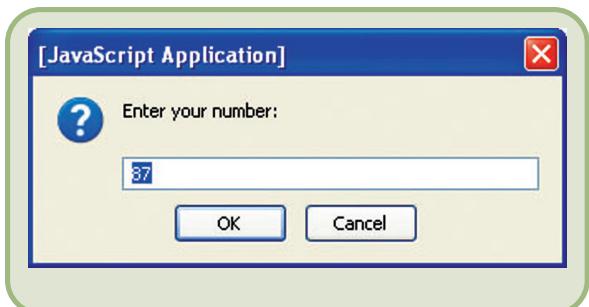
उक्त कथन इफ...एल्स कथन का और विस्तार है। यह कार्यान्वयन हेतु कथनों के कई सेटों में से चयन करने के लिए उपयोगी है।

इसका सिंटेक्स है–

```
if (condition1)
{
    code to be executed if condition1 is true
}
else if (condition2)
{
    code to be executed if condition2 is true
}
.
.
else
{
    code to be executed if any of the conditions is not true
}
```

*चित्र 10.12 – **इफ...एल्स इफ...एल्स** कथन का प्रवाहचित्र*

उदाहरण– यह जांच करने हेतु प्रोग्राम लिखें कि क्या प्रॉम्प्ट बॉक्स से इनपुट की गई संख्या शून्य, विषम अथवा सम है।

```
<HTML>
<HEAD>
    <TITLE>Odd, Even or Zero</TITLE>
</HEAD>
<BODY>
<SCRIPT type = “text/JavaScript”>
    var n = prompt(“Enter your number:”,”Type your number¿
here”);
    n = parseInt(n);         //converts string into number
    if (n == 0)
          alert(“The number is zero!”);
    else if (n%2)
          alert(“The number is odd!”);
    else
          alert(“The number is even!”);
</SCRIPT>
</BODY>
</HTML>
```

चित्र 10.13 – आउटपुट

आउटपुट

स्विच कथन से चयन

स्विच कथन का प्रयोग विभिन्न कंडीशनों पर आधारित विभिन्न कथनों का कार्यान्वयन करने के लिए किया जाता है। यह इफ...एल्स...इफ कथनों के लंबे क्रम का बेहतर विकल्प प्रदान करता है।

इसका सिंटेक्स है

switch (*expression*)
{
case *label1* : //executes when value of exp. evaluates to label
statements;
break;
case *label2* :
statements;
break;
...
default : *statements*; //executes when non of the above labels //matches the result of expression
}

प्रोग्राम 10.5– प्रॉम्प्ट डायलॉग बॉक्स में तिथि इनपुट द्वारा सप्ताह के दिनों की (शब्दों में) गणना करना।

```
<HTML>
<HEAD>
    <TITLE>Switch statement</TITLE>
</HEAD>
<BODY>
<script type="text/JavaScript">
// Enter date of birth to know the day of week on that day.
var d=new Date(prompt("Enter your Date of Birth
(e.g. November 17, 2002)", "Month DD, YYYY"))
dy=d.getDay()
switch (dy)
{
case 0:
    document.write("It was <b>Sunday</b> on that day.")
    break
case 1:
    document.write("It was <b>Monday</b> on that day.")
    break
case 2:
    document.write("It was <b>Tuesday</b> on that day.")
    break
```

```
case 3:
    document.write("It was <b>Wednesday</b> on that day.")
    break
case 4:
    document.write("It was <b>Thursday</b> on that day.")
    break
case 5:
    document.write("It was <b>Friday</b> on that day.")
    break
case 6:
    document.write("It was <b>Saturday</b> on that day.")
    break
default:
    document.write("Please input a
valid Date in?
prescribed format !!!")
}
</script>
</BODY>
</HTML>
```

[JavaScript Application]

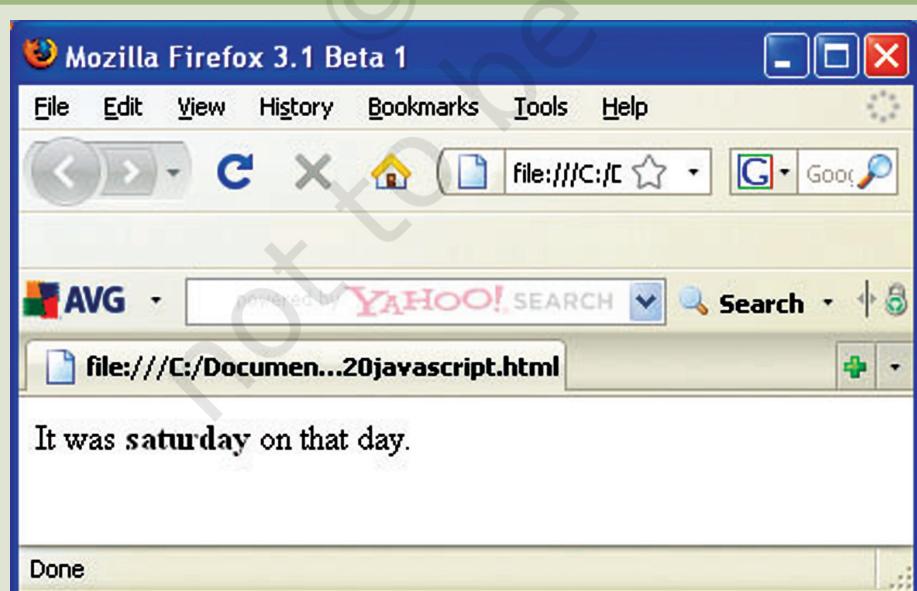
Enter your Date of Birth (e.g. November 17,2000)

October 5,2002

OK Cancel

तत्पश्चात् इस अभिव्यक्ति के मान की तुलना स्विच संरचना में प्रत्येक केस के मान से की जाती है। यदि यह समान होता है तो उस केस से संबद्ध कोड के ब्लॉक को कार्यान्वित किया जाता है। यदि केस समान नहीं होता तो डिफाल्ट कथन को कार्यान्वित किया जाता है। कोड को अगले केस में स्वत: जाने से रोकने के लिए **'ब्रेक'** का प्रयोग करें। डिफाल्ट केस में ब्रेक का प्रयोग करने की कोई आवश्यकता नहीं है।

जब आप 'ओके' बटन दबाते हैं तो आउटपुट प्राप्त होता है (चित्र 10.14)।

चित्र 10.14 – आउटपुट

10.10.2 लूप कथन

लूप कथन प्राथमिक तंत्र हैं जो जावास्क्रिप्ट इंटरप्रेटर को यह बताते हैं कि कथनों को तब तक बार-बार कार्यान्वित किया जाए जब तक कि एक विनिर्दिष्ट कंडीशन पूरी नहीं हो जाती। जावास्क्रिप्ट निम्न लूपिंग कथनों को सहयोग करता है–

- फॉर
- डू... व्हाइल
- व्हाइल लूप

अधिकांश लूपों में प्रति-चर होता है जिसे लूप आरंभ करने से पूर्व शुरू किया जाता है और तत्पश्चात् इसकी लूप के दोहराव से पहले मूल्यांकित कंडीशन (अभिव्यक्ति) भाग के रूप में जांच की जाती है। अंततः प्रति-चर को कंडीशन के पुनः मूल्यांकन से एकदम पूर्व लूप बॉडी के अंत में इंक्रीमेंटेड या अपडेटेड किया जाता है।

'फॉर' (For)

'फॉर' लूप में तीन वैकल्पिक अभिव्यक्तियां होती हैं जो सेमीकॉलन द्वारा पृथक होती हैं, तत्पश्चात् लूप में कार्यान्वित कथनों का ब्लॉक होता है। लूप कथनों को बार-बार दोहराया जाता है जब तक कि कंडीशन असत्य नहीं हो जाती। ***'फॉर'*** लूप का प्रयोग तब किया जाता है जब हम पहले से यह जानते हों कि स्क्रिप्ट कोड को कितनी बार चलाया जाना चाहिए।

इसका सिंटेक्स है

for([*initial-expression*]; [*condition*]; [*increment-expression*])
{
statements
}

पैरामीटर

पैरामीटर-अभिव्यक्ति – प्रति-चर को आरंभ करने हेतु प्रयुक्त।

कंडीशन – यदि कंडीशन का मूल्यांकन सत्य होता है तो कथन कार्यान्वित किए जाते हैं।

इंक्रीमेंट-अभिव्यक्ति – प्रति-चर को बढ़ाने (इंक्रीमेंट) हेतु।

उदाहरण

निम्न ***'फॉर'*** कथन, चर i को घोषित करते हैं और इसे 1 से आरंभ करते हैं। यह जांच करता है कि i, 20 से कम है तथा दो अनुवर्ती कथनों को कार्यान्वित करता है और लूप के माध्यम से प्रत्येक पास के पश्चात् i में 2 की वृद्धि करता है।

```
// for loop to display odd numbers between 1 to 20
for (var i = 1; i < 20; i+=2)
{
```

```
        document.write(i);
        document.write("<BR>");
    }
```

प्रोग्राम 10.6– *'फॉर लूप'* कथन के प्रयोग से संख्या 7 का पहाड़ा निकालने हेतु जावास्क्रिप्ट कार्यक्रम।

```
<HTML>
<HEAD>
    <TITLE> Table of 7 </title>
</HEAD>
<BODY>
<SCRIPT language="JavaScript" type="text/JavaScript">
    document.write("<H2> Table of number 7 </H2>");
    for(i = 1; i <= 10; i++ )
    {      document.write(7*i);
           document.write("<BR>");
    }
</SCRIPT>
</BODY>
</HTML>
```

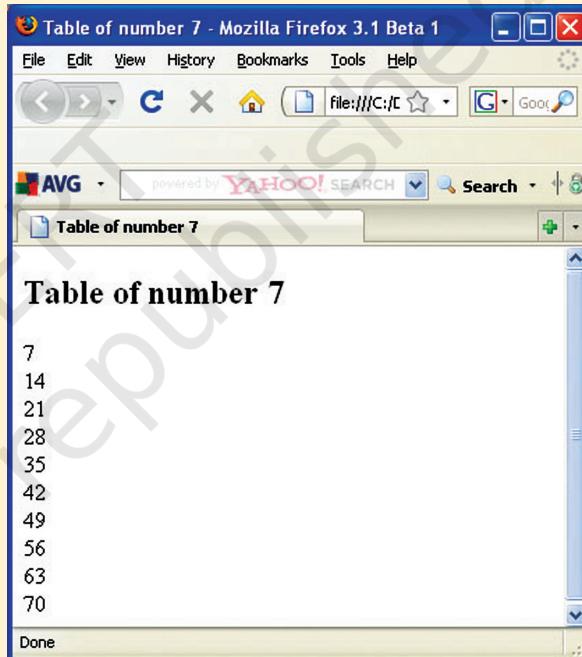
Table of number 7 - Mozilla Firefox 3.1 Beta 1

File Edit View History Bookmarks Tools Help

Table of number 7

Table of number 7

7
14
21
28
35
42
49
56
63
70

Done

व्हाइल (While)

व्हाइल लूप कथन, ***फॉर*** लूप से सरल होता है। इसमें वस्तुस्थिति (कंडीशन) एवं ब्लॉक कथन शामिल होता है। लूप द्वारा प्रत्येक बार गुज़रने से पहले कंडीशन का मूल्यांकन होता है। यदि कंडीशन सत्य है तो वह ब्लॉक कथन का कार्यान्वयन करती है।

इसका सिंटेक्स निम्नवत् है–

```
while (condition)
{
statements
}
```

उदाहरण: निम्नवत् व्हाइल लूप पूर्व उदाहरण में फॉर लूप के समान आउटपुट देता है।

```
// While loop to display Odd numbers between 1 to 20
var i = 1;      // Initialization of counter variable
while (i < 20)  // Condition
{
    document.write(i);
    document.write(“<BR>”);
    i++;         // Updation
}
```

व्हाइल लूप में, हम एक प्रतिचर को सुरक्षित करते हैं जो लूप में कथन का संपादन नियंत्रित करता है।

प्रोग्राम 10.7: प्रथम 10 Fibonacci संख्याएं निर्मित करने के लिए जावास्क्रिप्ट प्रोग्राम लिखें।

```
<HTML>
<HEAD>
<TITLE>Fibonacci Series</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE=”JavaScript” TYPE=”text/JavaScript”>
// Program to print the Fibonacci series upto 10 numbers
document.write(“Fibonacci Series... upto 10 numbers <BR>”.fontsize(4));
//.fontsize to increase the font size of the string
i=0;
document.write(i + “  “);
j=1;
document.writeln(j + “  “);
var x = 3;
while (x <= 10)
{
    t = i + j;
    document.write(t + “  “);
    i = j;
    j = t;
    x++;
}
</SCRIPT>
</BODY>
</HTML>
```


डू...व्हाइल (Do...While)

डू...व्हाइल लूप व्हाइल लूप के समान ही होता है। यह लूप को तब तक दोहराएगा जब तक निर्दिष्ट स्थिति गलत न हो। इस लूप को हमेशा कम से कम एक बार कार्यान्वित करना होगा,

यदि कंडीशन गलत भी हो तब भी, क्योंकि कंडीशन परीक्षण से पहले ही कथनों का ब्लॉक कार्यान्वित होता है। इस लूप कथन में, कोष्ठक चिह्न वैकल्पिक हैं।

इसका सिंटेक्स निम्नवत् है–

```
do
{
statements
}
while (condition);
```

उदाहरण: निम्नवत् डू...व्हाइल लूप समान आउटपुट देता है जैसा कि पूर्व उदाहरण में व्हाइल लूप ने दिया है।

```
// डू...व्हाइल लूप 1 से 20 तक के बीच की विषम संख्याओं को डिस्प्ले करता है।
var i = 1;  // प्रतिचर का आरंभीकरण
do
{
    document.write(i);
    document.write("<BR>");
    i++;  // अद्यतन
}
while (i < 20); // कंडीशन
```

व्हाइल कीवर्ड सहित कथन के ब्लॉक के पश्चात् कंडीशन कोष्ठक चिह्न के मध्य आता है।

10.10.3 लेबल

लेबल एक आइडेन्टिफायर है जो कि विसर्ग चिह्न द्वारा दर्शाया जाता है जिससे प्रोग्राम के प्रवाह को निर्देशित करने में सहायता मिलती है।

इसका सिंटेक्स निम्नवत् है–

लेबल: कथन

लेबल का मान कोई भी जावास्क्रिप्ट आइडेन्टिफायर हो सकता है। लेबल से आप जो कथन चिह्नित करते हैं वह कोई भी कथन हो सकता है।

उदाहरण

इस उदाहरण में, लेबल 'व्हाइल लूप' व्हाइल लूप को चिह्नित करता है।

```
x=1;
whileloop:            // Label
while(x<=10)
{
document.write(x);
x++;
}
```

10.10.4 ब्रेक

सबसे आंतरिक लूप, स्विच कथन या लेबल द्वारा नामित कथन से बाहर निकलने के लिए ब्रेक कथन का प्रयोग होता है। यह वर्तमान लूप को समाप्त करता है और समाप्त किए गए लूप कथन के बाद कथन को कंट्रोल कर अंतरित करता है।

इसका सिंटेक्स निम्नवत् है

break [*lable*]

ब्रेक कथन में एक वैकल्पिक लेबल होता है जिसमें कंट्रोल लेबल कथन से बाहर आ सकता है।

उदाहरणः निम्नवत् प्रोग्राम सेगमेंट में ब्रेक कथन है जो कि व्हाइल लूप को तब समाप्त करता है जब यह 3 के बराबर होता है।

```
var i = 0;
while (i <6)
{
if (i = = 3)
break;              // प्रथम दोहराव में कंट्रोल लूप से बाहर चला जाता है।
i + +;
}
document.write(i);
```

10.10.5 कंटीन्यू (Continue)

कंटीन्यू कथन आगे के कथन को छोड़ देता है और अगले दोहराव सहित लूप को कार्यान्वित करता है। इसे 'इफ' कथन के साथ व्हाइल, डू...व्हाइल फॉर या लेबल कथनों में प्रयोग किया जाता है।

इसका सिंटेक्स निम्नवत् है

continue [*lable*]

कंटीन्यू कथन लूप को समाप्त नहीं करता है। इसके बजाय, ***व्हाइल*** लूप में, यह कंडीशन पर वापिस आता है और ***फॉर*** लूप में यह अद्यतन अभिव्यक्ति पर आता है। ***कंटीन्यू*** कथन में वैकल्पिक लेबल हो सकता है जिससे प्रोग्राम विनिर्दिष्ट लेबल्ड कथन को ज़ारी रखता है और लेबल्ड कथन को समाप्त कर सकता है।

उदाहरण

एक प्रोग्राम को प्रॉम्प्ट ( ) के प्रयोग से 50 तत्वों का इनपुट देना है फिर कंटीन्यू कथन के प्रयोग से 40 से अधिक अंकों के जोड़ की गणना करनी है।

```
var marks = new Array();
var i = 0, sum=0;
while (i < 50)
{
```

```
        i++;
        // parseInt स्ट्रिंग मान को संख्या में बदलता है।
        marks[i]=parseInt(prompt("Enter marks"));
        if (marks[i] <= 40) // कंट्रोल व्हाइल कंडीशन अभिव्यक्ति पर जाता है।
        continue;    // control goes to while condition expression.
        sum = sum + marks[i];
    }
    document.write(sum+"\n");
```

10.11 वस्तु नियंत्रक कथन

जावास्क्रिप्ट कुछ वस्तु नियंत्रक विशिष्ट कथन होते हैं। ***फॉर... इन*** और ***विद्*** दो प्रकार के कथन हैं।

10.11.1 फॉर... इन

फॉर...इन कथन वस्तु की सभी विशेषताओं पर निर्दिष्ट चर को दोहराता है।

इसका सिंटेक्स निम्नवत् है–

for (<*variable*> in <*object*>)
{
statements
}

फॉर...इन कथन की विषय-वस्तु, वस्तु की प्रत्येक विशेषता के लिए एक बार कार्यान्वित की जाती है। लूप कथन के कार्यान्वयन से पहले वस्तु की एक विशेषता का नाम स्ट्रिंग के रूप में चर को सौंपा जाता है। हम [ ] ऑपरेटर के साथ वस्तु विशेषता के मान को देखने के लिए इस चर का प्रयोग कर सकते हैं।

उदारहण: बुक वस्तु की विशेषता के नाम और मान को प्रिंट करना।

```
<HTML>
<HEAD>
<TITLE> for... in Example </TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript" TYPE="text/javascript">
var Book = new Object();          // Object creation
// Properties and values of Book
Book =    {     Title:"The Discovery of India",↵
          Author:"Jawahar Lal Nehru", ↵
          Publisher: "Penguin Books", ↵
          Price:Rs 399//- ↵
    };
var result = "";
// Name of distinct property of Book assign to b in each loop
execution
for (var b in Book)
{
    // Book[b] is used to get the values.
    result += "Book." +  b + " = " + Book[b] + "<br>";
```

```
}
// To print names and values of each property of Book object.
document.write(result);
</SCRIPT>
</BODY>
</HTML>
```

आउटपुट

चित्र 10.17 – फॉर...इन कथन का प्रयोग

10.11.2 विद् (With)

विद् कथन तब अधिक से अधिक टाइपिंग से बचाता है जब समान वस्तु की विशेषता को जानना होता है। उदाहरण के लिए सघन नेस्टेड ऑब्जेक्ट क्रम के साथ कार्य करना सामान्य बात है। कई बार हमें HTML फॉर्म के तत्वों तक पहुँचने के लिए निम्नवत् अभिव्यक्ति को टाइप करना पड़ता है।

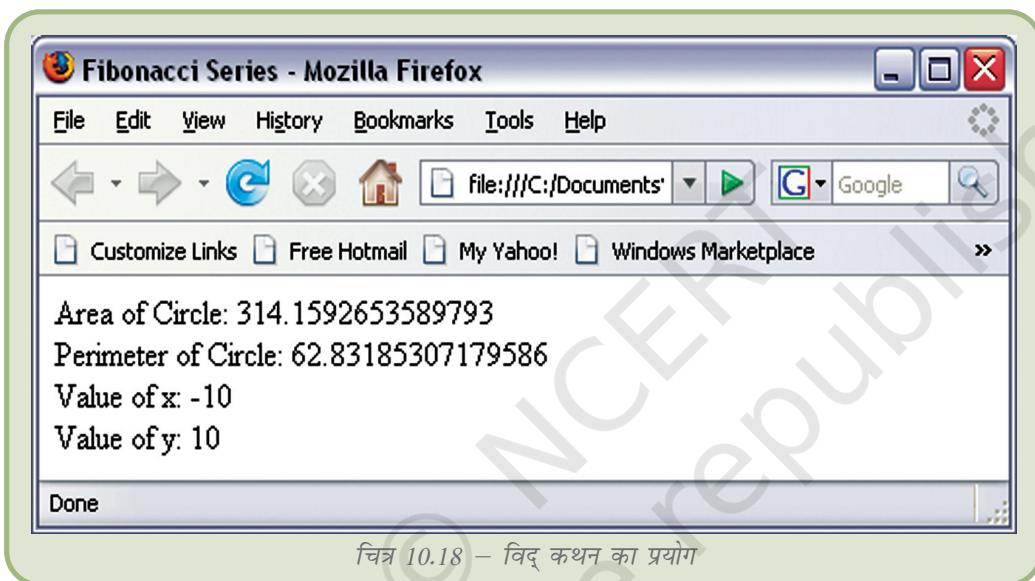
Frames[1].documents.forms [0].fname.value

उदाहरण

निम्नवत् स्क्रिप्ट ***विद्*** कथन के प्रयोग की व्याख्या करती है। यहाँ गणित (मैथ) और प्रलेख दो डिफॉल्ट वस्तु हैं। मैथ वस्तु के साथ ***विद्*** कथन के बाद का कथन वस्तु को विनिर्दिष्ट किए बिना, Cos और sin विधि एवं PI विशेषता को निर्दिष्ट करता है। इसी प्रकार से प्रलेख वस्तु के साथ ***विद्,*** कथन के बाद का कथन वस्तु को विनिर्दिष्ट किए बिना, लेखन विधि को निर्दिष्ट करता है। जावास्क्रिप्ट इन संदर्भों हेतु गणित और प्रलेख वस्तु को समझ लेती है।

```
var area, circumference
var r=10
with (Math)
{
   area = PI * r * r
   circumference  = 2*PI*r
}
with (document)
{
    write("Area of the Circle: "+area+"<br>");
    write("Circumference of the Circle: "+circumference);
}
```

आउटपुट

चित्र 10.18 – विद् कथन का प्रयोग

10.12 जावास्क्रिप्ट फंक्शन

फंक्शन कथनों का नामित ब्लॉक है जिसे सीधे नाम लिखकर बार-बार कार्यान्वित किया जा सकता है और यह कुछ मान प्रदान करता है। यह प्रोग्राम का मॉड्यूलर और समझने योग्य बनाने में उपयोगी है।

10.12.1 फंक्शन परिभाषित करना

निम्नवत् सिंटेक्स के प्रयोग से फंक्शन को परिभाषित कर सकते हैं–

इसका सिंटेक्स निम्नवत् है–

```
function  <function-Name>([<parameter list>])
{
  ... body of the function ..
}
```

फंक्शन की परिभाषा कीवर्ड फंक्शन से आरंभ होती है और इसके बाद कोष्ठक के अंतर्गत वैकल्पिक पैरामीटर-सूची और फंक्शन का नाम लिखा जाता है। फंक्शन में सभी कथनों को रखने के लिए ({और}) कोष्ठकों का प्रयोग होता है। आइए फंक्शन की परिभाषा का उदाहरण देखें–

```
function Welcome()
{
    alert("Welcome to NCERT ");
}
```

ये कथन वेलकम फंक्शन को परिभाषित करता है जो प्रयोगकर्ता के लिए अलर्ट मैसेज को डिसप्ले करता है।

10.12.2 फंक्शन के साथ पैरामीटर का प्रयोग

समतुल्य मानों की सूची में फंक्शन के तर्कों को पैरामीटर कहते हैं। ये मान फंक्शन के अंतर्गत स्थानीय चर को दिए जा सकते हैं। आइए पैरामीटर के प्रयोग से उदाहरण देखें–

```
function Welcome(name)
{
    alert("Welcome to NCERT , " + name);
}
```

हमने सीखा कि सामान्य फंक्शन कैसे निर्मित किए जाते हैं। फंक्शन की परिभाषा हेतु सबसे उपयुक्त स्थान HTML प्रलेख के <HEAD> खंड में है, क्योंकि इस खंड में पहले कथनों को कार्यान्वित किया जाता है जो यह सुनिश्चित करता है कि फंक्शन को प्रयोग से पहले परिभाषित करते हैं।

उदाहरण– HTML प्रलेख के HEAD खंड के अंतर्गत फंक्शन का प्रयोग करना।

```
<HTML>
<HEAD>
<TITLE> Define a Function </title>
<script language="JavaScript" type="text/JavaScript">
function Welcome(name)
{
    alert("Welcome to NCERT, " + name+"!");
}
</script>
</HEAD>
<BODY>
This is the body of HTML document. You will observe that I
have used script code in small case and other HTML tags in
Capital to enhance the redability of JavaScript Code.
</BODY>
</HTML>
```

10.12.3 फंक्शन बुलाना

फंक्शन को तर्कों की सूची के साथ फंक्शन का नाम लिखकर किया जा सकता है। फंक्शन कॉल को इवेंट हैंडलर कोड में भी प्रयोग किया जा सकता है।

इसका सिंटेक्स है–

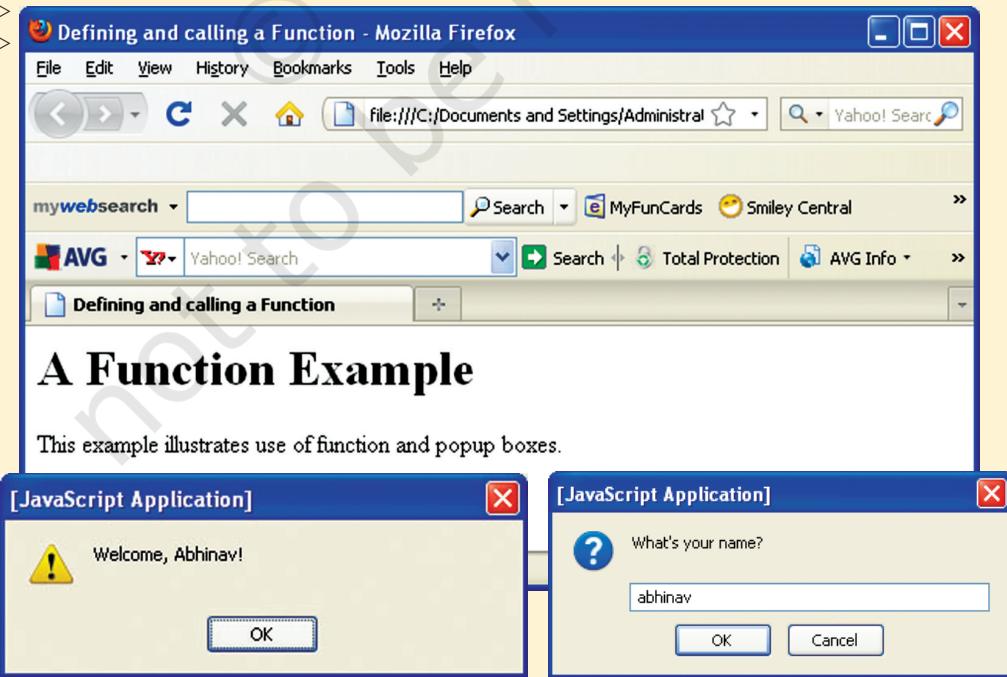
<function name> ([<parameter list>])

उदाहरण

```
Welcome("MANYA");
```

यह जावास्क्रिप्ट इंटरप्रेटर से 'Welcome' फंक्शन के प्रथम कथन में कंट्रोल को अंतरित करने के लिए कहता है। यह **"MANYA"** पैरामीटर को भी फंक्शन भेजता है। फंक्शन के भीतर ***नेम*** चर को मान निर्दिष्ट किया जाएगा।

प्रोग्राम 10.8 – HTML प्रलेख फंक्शन की परिभाषा और वेलकम ( ) फंक्शन को आरंभ करने की व्याख्या करता है।

```
<HTML>
<HEAD>
<TITLE> Defining and calling a Function </title>
<script language="JavaScript" type="text/JavaScript">
// function definition
function Welcome(name)
{
    alert("Welcome, " + name);
}
</script>
</HEAD>
<BODY>
<H1> A Function Example</H1>
This example illustrates use of function and popup boxes.
<br>
<script language="JavaScript" type="text/JavaScript">
    var nm = prompt("What's your name? ", "Your name please…");
    // Calling of function Welcome().
    Welcome(nm);
</script>
</BODY>
</HTML>
```

10.12.4 एक इवेंट पर फंक्शन बुलाना

एक बार फंक्शन को परिभाषित किए जाने के बाद इसे क्लिक इवेंट जैसे इवेंट के साथ प्रयोग किया जा सकता है। यहाँ पर फंक्शन जावास्क्रिप्ट की एक कमांड बन जाती है। उदारहण के लिए–

```
<INPUT type = "button"
   value = "Calculate"
   onClick = calc() >
```

जब प्रयोगकर्ता बटन पर क्लिक करता है, प्रोग्राम स्वत: ही calc () फंक्शन आरंभ कर देता है।

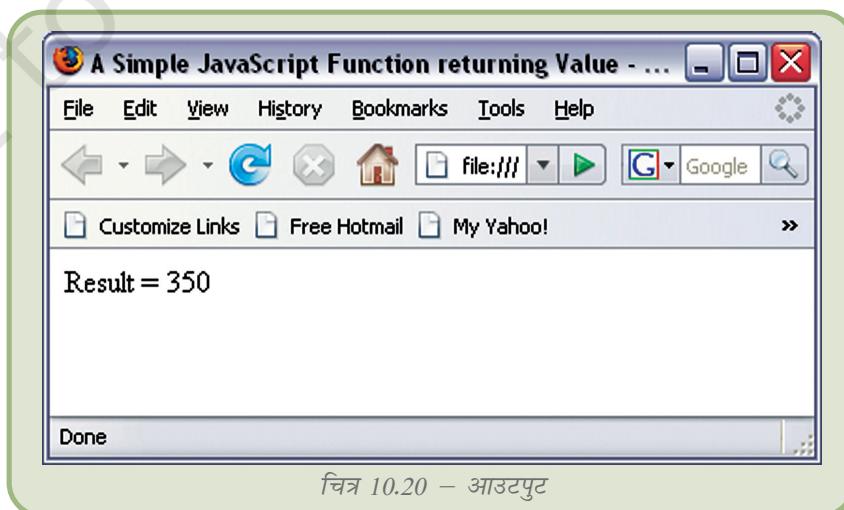
10.12.5 फंक्शन से मान प्राप्त करना

रिटर्न कथन फंक्शन से मान प्राप्त करने के लिए प्रयोग होता है। एसाइनमेंट ऑपरेटर का प्रयोग करने वाला चर रिटर्न मान को धारण कर सकता है।

उदारहण के लिए, फंक्शन के प्रयोग से साधारण ब्याज की गणना करने हेतु प्रोग्राम–

```
<HTML>
<HEAD>
<TITLE>A Simple JavaScript Function returning Value </TITLE>
<script language="JavaScript" type="text/JavaScript">
    function si( p, r, t )
    {      var s = (p * r * t)/ 100
           return s;    // function returning value s
    }
</script>
</HEAD>
<BODY>
<script language="JavaScript" type="text/JavaScript">
    var result = si( 1000, 5, 7)   // returned value assigned
to result
    document.write ( "Result = " + result);
</script>
</BODY>
</HTML>
```

आउटपुट

चित्र 10.20 – आउटपुट

10.12.6 नेस्टेड फंक्शन

फंक्शन को अन्य फंक्शन परिभाषा के अंदर रखा जा सकता है। ये (इफ कथन या व्हाइल लूप इत्यादि की विषय–वस्तु के रूप में) कथन ब्लॉक के अंतर्गत परिभाषित नहीं किए जा सकते हैं।

मान लो कि हम वृत के क्षेत्रफल की गणना करना चाहते हैं, जो कि $3.14 \times r^2$ के बराबर है। इसके लिए नेस्टेड फंक्शन निम्नवत् रूप से लिखा जाएगा–

```
function Area(r)
{
   function Square(x)
   {
         return x*x;
   }
   return 3.14*Square(r);
}
```

इस स्क्रिप्ट में, वर्ग () फंक्शन क्षेत्रफल () फंक्शन के अंदर नेस्टेड होता है।

जावास्क्रिप्ट में आरक्षित / मुख्य शब्द

एब्सट्रैक्ट	बूलियन	ब्रेक	बाइट
केस	कैच	चार	क्लास
काँस्ट	कंटीन्यू	डीबगर	डिफॉल्ट
डिलीट	डू	डबल	एल्स
एनम	एक्सपोर्ट	एक्सटेंड्स	फाल्स
फाइनल	फाइनली	फ्लोट	फॉर
फंक्शन	गोटू	इफ	इम्पलीमेंट
इम्पोर्ट	इन	इन्सटांस ऑफ	इन्ट
इंटरफेस	लॉन्ग	नेटिव	न्यू
नल	पैकेज	प्राइवेट	प्रोटेक्टेड
पब्लिक	रिटर्न	शॉर्ट	स्टेटिक
सुपर	स्विच	सिंक्रोनाइज़्ड	दिस
थ्रो	थ्रोज	ट्रॉजिएन्ट	ट्रू
ट्राई	टाइपऑफ	वार	वोइड
वोलाटाइल	व्हाइल	विद्	

सारांश

- जावास्क्रिप्ट एक मंच जो स्वतंत्र वस्तु - आधारित स्क्रिप्टिंग लैंग्वेज है।
- क्लांइट-साइड जावास्क्रिप्ट, जावास्क्रिप्ट कोड को दिया गया नाम है जो क्लांइट मशीन पर वेब ब्राउज़र द्वारा कार्यान्वित होता है।
- जावास्क्रिप्ट स्थिति अनुकूल (केस सेंसिटिव) भाषा है और सभी कथन छोटे अक्षरों (लोअर केस) में लिखे जाते हैं।
- जब कथनों को अलग-अलग पंक्तियों में रखा जाता है, जावास्क्रिप्ट द्वारा अर्धविराम को हटाया जा सकता है। यदि हम एक पंक्ति में कथनों का मेल करते हैं तो हमें उन्हें अलग-अलग करने के लिए अर्धविराम का प्रयोग करना चाहिए।
- ***document.write*** प्रलेख विंडो से आउटपुट प्राप्त करने के लिए मानक जावास्क्रिप्ट कमांड है।
- लिटरल्स उन स्थिर मानों को निर्देशित करता है जिनका प्रोग्राम कोड में प्रत्यक्ष रूप से प्रयोग होता है।
- चर मान या स्ट्रिंग का कंटेनर है। चर में सुरक्षित मानों को चर के नाम के प्रयोग से प्राप्त किया जा सकता है।
- जावास्क्रिप्ट तीन पुराने डाटा प्रकार को समर्थन करती है– संख्या, स्ट्रिंग और बूलियन। जावास्क्रिप्ट वस्तु और विन्यास दो कंपोजिट डाटा प्रकार को समर्थन करता है।
- अभिव्यक्ति ऑपरेटर और लिटरल्स या चर नामों का मेल है।
- इंक्रीमेंट और डिक्रीमेंट ऑपरेटर केवल एक ऑपरैंड पर कार्य करते हैं। 1 द्वारा चर का मान हटाने या बढ़ाने के लिए इनका प्रयोग होता है।
- असाइनमेंट बेसिक ऑपरेटर है, जो इसके दाएं ऑपरैंड के मान को इसके बाएं ऑपरैंड को देता है। इस ऑपरटेर को बराबर के चिह्न (=) द्वारा प्रस्तुत किया जाता है।
- संबंधात्मक ऑपरेटरों का ऑपरैंड के मान की तुलना करने के लिए प्रयोग होता है और यह स्थिति के आधार पर बूलियन मान को दर्शाता है।
- जावास्क्रिप्ट के तीन तार्किक ऑपरेटर **हैं: &&** (AND), || (OR), तथा ! (NOT)
- + ऑपरेटर दो स्ट्रिंग ऑपरैंड को शृंखलाबद्ध करने के लिए भी प्रयोग होता है। यह अंकीय ऑपरैंड की तुलना में स्ट्रिंग ऑपरैंड को अधिक महत्त्व देता है।
- कंडीशनल ऑपरेटर (?:) को टरनरी ऑपरेटर भी कहा जाता है।
- प्रयोगकर्ता-परिभाषित वस्तु प्रकार या पूर्व परिभाषित वस्तु प्रकारों में से एक के निर्माण के लिए **न्यू** ऑपरेटर का प्रयोग होता है।
- मैमोरी स्पेस को रिक्त करने के लिए ***डिलीट्*** ऑपरेटर का प्रयोग होता है।
- यदि विनिर्दिष्ट वस्तु में निर्दिष्ट विशेषता/सूची है तो 'इन' ऑपरेटर सही दर्शाता है।
- अलर्ट बॉक्स ''ओके'' बटन और टेक्स्ट मैसेज सहित डायलॉग बॉक्स है।
- कन्फर्म बॉक्स कुछ सूचनाओं का सत्यापन या स्वीकृत करने हेतु है, प्रयोगकर्ता को आगे बढ़ने के लिए या तो ''ओके'' या ''कैंसल'' बटन पर क्लिक करना होता है।

- प्रॉम्प्ट बॉक्स से हम टेक्स्ट फील्ड की सहायता से प्रयोगकर्ता का इनपुट एकत्र कर सकते हैं।
- ब्लॉक कथन दो या अधिक कथनों को एक कथन में जोड़ता है। सामान्यतया ब्लॉक कथनों का कंडीशनल और लूपिंग कथनों के साथ प्रयोग होता है।
- कंडीशन के आधार पर कथन या ब्लॉक कथनों को कार्यान्वित करने के लिए ***'इफ'*** कथन का प्रयोग करता है।
- जावास्क्रिप्ट में स्विच कथन का प्रयोग विभिन्न कंडीशनों पर आधारित विभिन्न कार्यों को प्रस्तुत करने के लिए किया जाता है। यह बहुत यदि...अन्यथा यदि...कथन का प्रतिस्थापक भी हो सकता है।
- लूप कथन जावास्क्रिप्ट इंटरप्रेटर को कुछ कथनों को तब तक बार-बार कार्यान्वित करने के लिए कहता है जब तक कि निर्दिष्ट स्थिति प्राप्त न हो।
- ***फॉर*** लूप में तीन वैकल्पिक अभिव्यक्तियाँ होती हैं। जब तक कंडीशन गलत नहीं होता है यह ब्लॉक कथनों को बार-बार दोहराता है।
- ***व्हाइल लूप कथन फॉर*** लूप से सरल होता है। जब तक निर्दिष्ट कंडीशन गलत नहीं होती है यह ब्लॉक कथन को बार-बार दोहराता है।
- ***डू...व्हाइल*** लूप हमेशा कम से कम एक बार कोड ब्लॉक को कार्यान्वित करता है जो कि ***व्हाइल*** लूप के समान नहीं है।
- लूप से बाहर निकलने की बजाय, कंटीन्यू कथन अनुवर्ती कथन को छोड़ देता है और नए दोहराव से लूप आरंभ करता है।
- ***फॉर...इन*** कथन वस्तु की सभी विशेषताओं पर विनिर्दिष्ट चर को दोहराता है।
- ***विद्*** कथन कथनों के सेट हेतु डिफॉल्ट वस्तु को स्थापित करता है।
- फंक्शन जावास्क्रिप्ट कथनों के समूह हेतु नेम्ड (नामित) इकाई है। यदि प्रयोगकर्ता को फंक्शन को मान भेजना है तो फंक्शन के नाम के पश्चात् मान को कोष्ठक में रखा जाता है और जब फंक्शन आरंभ करते हैं तो इसे कौमा लगी हुई तर्कों की सूची के रूप में भेजता है।
- ***रिटर्न*** कथन का प्रयोग फंक्शन से मान प्राप्त करने के लिए होता है। असाइनमेंट (ऑपरेटर का प्रयोग करने वाला चर) दर्शाए गए मान को धारित कर सकता है।
- फंक्शन को अन्य फंक्शन परिभाषा के अंदर नेस्टेड किया जा सकता है

अभ्यास

लघु उत्तर वाले प्रश्न

1. क्या जावास्क्रिप्ट में एकल अनुप्रयोग हेतु प्रोग्राम लिखना संभव है?
2. हम कथनों में अर्धविराम को कहाँ पर प्रयोग करते हैं? क्या यह अनिवार्य है?
3. क्लाइंट-साइड और सर्वर-साइड जावास्क्रिप्ट में अंतर स्पष्ट करें।
4. जावास्क्रिप्ट कोड में ***document.write()*** का क्या उद्देश्य है?

5. निम्नवत् में अवैध चरों के नाम क्या हैं? कारण सहित व्याख्या करें। My_Name, number 10, $ 100, father's name, marks %
6. आरक्षित शब्द क्या हैं? क्या हम आरक्षित शब्दों को आइडेन्टिफायर्स के रूप में प्रयोग कर सकते हैं?
7. लिटरल्स का क्या अर्थ है?
8. जावास्क्रिप्ट में डाटा प्रकार क्या हैं?
9. ***var*** कथन का क्या उद्देश्य है?
10. एकल या दोहरे कोट्स (quotes) के अंतर्गत स्ट्रिंग क्यों रखे जाते हैं?
11. निम्नवत् को दर्शाने के लिए जावास्क्रिप्ट कोड लिखें–

    He said, "Mahatma Gandhi was a non-violent soldier of India."
12. क्या हम एकल विन्यास में विभिन्न डाटा प्रकार के मानों का प्रयोग कर सकते हैं?
13. ***stumarks*** नामक चर को 65, 75, 80, 87, 90 अंकीय मान देने के लिए कथन लिखें।
14. क्या नल और शून्य का समान मान है?
15. ऑपरेटर क्या होते हैं? ऑपरेटर का क्या महत्त्व है?
16. 'प्रॉम्प्ट( ) विधि किस प्रकार के मान को दर्शाता है?
17. निम्नवत् हेतु समकक्ष जावास्क्रिप्ट कथन लिखें–

    (क) क्षेत्रफल = $3.14r^2$

    (ख) KE = $1/2\ mv^2$
18. चर **r** का मान क्या होगा?
    ```
    var r = 45 % 7;
    ```
19. चर r का मान क्या होगा?
    ```
    var a = 4;
    var b = 7;
    var c = "NCERT";
    r = a + b + c;
    ```
20. '= =' ऑपरेटर असाइनमेंट ऑपरेटर ('=') के समान नहीं है (सही या गलत)।
21. पूर्ण चरों का मान ज्ञात करें।

    कुल (360 * 5) + ((40 / 8) - 9) - ((14 * 6) / 2);
22. चर b का मान क्या है?
    ```
    a = 25;
    b = (+ + a) + 7;
    ```

दीर्घ उत्तर वाले प्रश्न

1. दिया गया वर्ष अधिवर्ष है या नहीं, यह जानने के लिए जावास्क्रिप्ट लिखें। वर्ष इनपुट के लिए प्रॉम्प्ट बॉक्स का प्रयोग करें।

2. निम्न द्वारा प्रयोगकर्ता का स्वागत करने वाले जावास्क्रिप्ट प्रोग्राम को लिखें–
   गुड मॉर्निंग, <यूजर_नेम> 4 से 11 के बीच के समय में
   गुड नून, <यूजर_नेम> जब बारह बजे हों
   गुड आफ्टरनून, <यूजर_नेम> जब 13 से 16 के बीच का समय हो
   गुड ईवनिंग, <यूजर_नेम> जब 17 से 23 के बीच का समय हो
   गुड नाइट, <यूजर_नेम> जब 1 से 3 या 24 के बीच का समय हो।
3. दी गई तीन संख्याओं में से सबसे बड़ी संख्या ज्ञात करने के लिए प्रोग्राम लिखें।
4. जब निम्नवत् जावास्क्रिप्ट कोड को कार्यान्वित किया जाता है तो क्या होता है?

```
var get_res = confirm("Did you like this chapter!?");
if (get_res = = true)
alert ("okay! Let's go to the next!");
```

5. यदि चर का ***अंक*** 75 है तो आउटपुट क्या होगा?

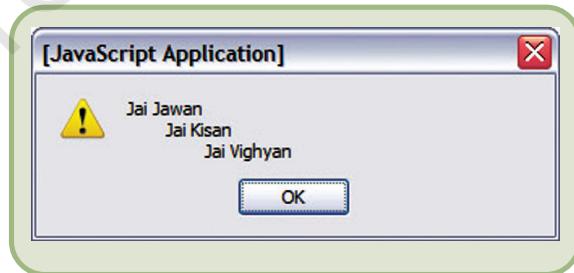
```
if  (marks > = 80)
   {
alert  ("Excellent!");
   }
else if (marks > = 60 && marks < 80)
   {
alert  ("Good!");
   }
else if  (marks > 50 && marks  <60)
   {
alert  ("Averaget!");
   }
else
   {
alert  ("Improve yourself !");
   }
```

6. 0 से 9 तक की संख्या के समकक्ष शब्द प्रिंट करने के लिए स्विच कथन के प्रयोग से प्रोग्राम लिखें। उदाहरण 3 को ''Three'' दर्शाना चाहिए।
7. अलर्ट बॉक्स को डिस्प्ले करने वाला कथन लिखें जो निम्नवत् दिखे–

[JavaScript Application]
Jai Jawan
Jai Kisan
Jai Vighyan
OK

8. ***'फॉर'*** लूप के प्रयोग से निम्नवत् आउटपुट को प्रिंट करने के लिए प्रोग्राम लिखें।

   1
   22
   333
   4444
9. निम्नवत् 'फॉर' लूप को कितनी बार कार्यान्वित किया जाएगा?

```
for  (a = 0; a <= 10; a = a + 2)
{
... statement ...
}
```

10. **'जोड़'** चर का अंतिम मान क्या होगा?
```
var sum = 0;
function add ( )
{
var sum = sum + 20;
}
add( );
sum = sum + 1;
```
11. स्क्रिप्ट कार्यान्वयन के अंत में अलर्ट बॉक्स में क्या डिस्प्ले होगा?
```
var y = 0;
for (x = 0; x <=  5; x++, y = y + 50 )
{
y = y + 10;
}
alert("The value of y is :" + Y);
```
12. विजिटर द्वारा प्रॉम्प्ट के माध्यम से प्रविष्ट की गई 5 संख्याओं के औसत की गणना करने हेतु प्रोग्राम लिखें।
13. यदि समय चर 'num' का **मान** 30 है, तो निम्नवत् व्हाइल लूप को कितनी बार कार्यान्वियत किया जाएगा?
```
while num <= 30 )
{
... statement ...
num = num + 3;
}
```
14. किसी संख्या का प्रतिलोम (अर्थात् 123 का प्रतिलोम 321 है) निकालने के लिए प्रोग्राम लिखें।
15. दशमलव संख्या को युग्मक संख्या में बदलने के लिए प्रोग्राम लिखें।
16. संख्या पैलिनड्रोम (palindrome) है या नहीं, यह जाँचने हेतु प्रोग्राम लिखें। एक संख्या तब ही पैलिनड्रोम होती है यदि वह अपनी प्रतिलोम संख्या के बराबर हो।
17. 1 से 20 के बीच की विषम संख्याओं को डिस्प्ले करने हेतु कंटीन्यू का प्रयोग करें।
18. किसी संख्या के अंकों का जोड़ उदाहरण 453 का परिणाम है 4 + 5 + 3 = 12 निकालने हेतु प्रोग्राम लिखें।
19. एक विशिष्ट सीमा तक प्राइम संख्या निकालने के लिए प्रोग्राम लिखें।
20. निम्नवत् जावास्क्रिप्ट कोड का आउटपुट क्या होगा?
```
for(i=1; i <=5; i++)
{
document.write("<BR>")
for(j=1; j <= i; j++)
document.write("*")
}
```
21. निम्नवत् कोड सैगमेंट में त्रुटियाँ ढूँढ़ें–
```
function 3_alert_box
{
alert ("Hi!, I am from a function.');
}
```

22. प्रथम 'n' संख्याओं का जोड़ जानने के लिए व्हाइल कथन को प्रयोग करते हुए प्रोग्राम लिखें?
23. निम्नवत् कोड सैगमेंट में त्रुटियाँ ढूँढ़ें–
```
fun_alert_box
document.write("Hi!, I am from a function");+1
}
```
24. नेस्टेड फंक्शन से क्या अभिप्राय है?

बहु वैकल्पिक प्रश्न

1. बाह्य जावास्क्रिप्ट हेतु फाइल एक्सटेंशन है–
   (i) .jscript
   (ii) .js
   (iii) .jav
   (iv) .java
2. जावास्क्रिप्ट प्रोग्रामों के अंतर्गत कमेंट का प्रयोग क्यों होता है?
   (i) ब्राउज़र को बताने के लिए जावास्क्रिप्ट कोड में HTML है।
   (ii) क्योंकि HTML के पास कमेंट हेतु अपनी कोई कमांड नहीं है।
   (iii) स्क्रिप्ट क्या करती है इसकी व्याख्या करने के लिए।
   (iv) उपर्युक्त में से कोई भी नहीं है।
3. जावास्क्रिप्ट का वास्तविक नाम था–
   (i) Javascript
   (ii) Livescript
   (iii) Wirescript
   (iv) ECMAScript
4. निम्नवत् में से कौन सी वैध अभिव्यक्ति नहीं है?
   (i) 7.5
   (ii) b + a = c
   (iii) {feet:10, inches:5}
   (iv) [2,3,6,9,5,7]
5. निम्नवत् अभिव्यक्ति में **res** का मान क्या होगा?
   var res = "70" + 25
   (i) 95
   (ii) 7025
   (iii) 25
   (iv) त्रुटि के कारण कोई आउटपुट नहीं है।
6. "India" <"bharat"> अभिव्यक्ति द्वारा दर्शाया गया मान क्या होगा?
   (i) 0
   (ii) सही
   (iii) गलत
   (iv) अवैध अभिव्यक्ति

7. ***डिलीट्*** ऑपरेटर का प्रयोग किसे डिलीट् (लोप) करने के लिए होता है?
   (i) .js files
   (ii) cookies
   (iii) निर्मित वस्तुएँ
   (iv) फंक्शन
8. किस वस्तु से कन्फर्म() विधि संबंधित है?
   (i) प्रलेख
   (ii) विंडो
   (iii) फ्रेम
   (iv) दिनांक
9. कथनों के ब्लॉक के कार्यान्वयन को दोहराने के लिए निम्नलिखित में से कौन-सा कथन प्रयोग होता है?
   (i) यदि ... अन्यथा ... (if ... else ..)
   (ii) कंटीन्यू
   (iii) व्हाइल
   (iv) स्विच
10. कथन के कार्यान्वयन के पश्चात् जोड़ का मान क्या होगा?
   for (i =1, sum = o; i + +) sum + = i;
   (i) 13
   (ii) 15
   (iii) 25
   (iv) कथन की त्रुटि
11. कोड का ब्लॉक कम से कम एक बार कार्यान्वयन कौन-सा लूप करता है?
   (i) व्हाइल
   (ii) फॉर
   (iii) डू...व्हाइल
   (iv) फॉर / इन
12. वस्तु की सभी विशेषताओं में विनिर्दिष्ट चर दोहराने के लिए निम्नवत् कौन-सा लूप कथन प्रयोग होता है?
   (i) फॉर
   (ii) डू ... व्हाइल
   (iii) फॉर / इन
   (iv) उपर्युक्त में से कोई भी नहीं

क्रियाकलाप

- जावास्क्रिप्ट कोड के प्रयोग HTML से प्रलेख लिखें जो आप का नाम, कक्षा और विद्यालय का नाम डिस्प्ले करता है।
- अपने वेब पेज में यादृच्छिक रूप से बैकग्राउंड के रंग को बदलने के लिए, जावास्क्रिप्ट के प्रयोग से HTML प्रलेख लिखें। (संकेत– document.bg color विशेषता और Math.random() फंक्शन का प्रयोग करें।)

- प्रयोगकर्ता मतदान करने योग्य है या नहीं यह जानने के लिए प्रोग्राम लिखें। प्रयोगकर्ता द्वारा मत देने हेतु निम्नवत् दो परिस्थितियों की जाँच की जानी चाहिए–
  (क) आयु 18 वर्ष या इससे अधिक होनी चाहिए।
  (ख) राष्ट्रीयता भारतीय होनी चाहिए।
- एक सामान्य क्विज प्रोग्राम लिखें जिसमें प्रयोगकर्ता से 5 प्रश्न पूछे जाएं। प्रश्नों के उत्तर के बारे में अलर्ट रहें और प्रतियोगिता के अंत में प्रयोगकर्ता को अंक दें।
- अपनी वेबसाइट में फॉर्म वैधता हेतु जावास्क्रिप्ट कोड के प्रयोग से HTML प्रलेख लिखें।

संदर्भ

जावास्क्रिप्ट – द डेफिनिटिव गाइड – डेविड फ्लैनेगन, ओ'रियली

सैम टीच् योरसैल्फ जावास्क्रिप्ट इन 24 आवर्स – माइकल मोकर, सैम प्रकाशन

जावास्क्रिप्ट-ए बिगनर्स गाइड – जॉन पॉलाक, मैकग्रॉ–हिल

http://www.webreference.com/programming/javascript
http:// www.webdevelopersnotes.com/tutorials/javascript
http://www.javascriptkit.com
http://www.w3schools.com/js
http://sun.com
http://www.mozilla.org